AI Benchmark Democratization and Carpentry

Contents

AI Benchmark Democratization and Carpentry

Gregor von Laszewski[1], Wesley Brewer[2], Jeyan Thiyagalingam[3], Juri Papay[3], Armstrong Foundjem[4]

Piotr Luszczek[5], Murali Emani[6], Shirley V. Moore[7], Vijay Janapa Reddi[8], Matthew D. Sinclair[9]

Sebastian Lobentanzer[10], Sujata Goswami[11], Benjamin Hawks[12], Marco Colombo[13], Nhan Tran[12]

Christine R. Kirkpatrick[14], Abdulkareem Alsudais[15], Gregg Barrett[16], Tianhao Li[17], Kirsten Morehouse[18]

Shivaram Venkataraman[9], Rutwik Jain[9], Kartik Mathur[20], Victor Lu[21], Tejinder Singh[22], Khojasteh Z. Mirza[23],

Kongtao Chen[24], Sasidhar Kunapuli[25], Gavin Farrell[26], Renato Umeton[27], Geoffrey C. Fox[1]

[1] *Biocomplexity Institute, University of Virginia*, Charlottesville, VA, USA
(laszewski@gmail.com, gcfexchange@gmail.com)
[2] *Oak Ridge National Laboratory*, Oak Ridge, TN, USA (brewerwh@ornl.gov)
[3] *Rutherford Appleton Laboratory, STFC*, Harwell Campus, UK
(t.jeyan@stfc.ac.uk, juri.papay@stfc.ac.uk)
[4] *DEEL, Polytechnique Montreal*, Montreal, Canada (a.foundjem@polymtl.ca)
[5] *LLSC, MIT Lincoln Laboratory*, Lexington, MA, USA (luszczek@icl.utk.edu)
[6] *Argonne National Laboratory*, Lemont, IL, USA (memani@anl.gov)
[7] *Computer Science Department, UTEP*, El Paso, TX, USA (svmoore@utep.edu)
[8] *Harvard University*, Boston, MA, USA (vj@eecs.harvard.edu)
[9] *Computer Sciences Department, Univ. of Wisconsin–Madison*, Madison, WI, USA
(sinclair@cs.wisc.edu, shivaram@cs.wisc.edu, rnjain@wisc.edu)
[10] *Helmholtz Center Munich*, Munich, Germany
(sebastian.lobentanzer@helmholtz-munich.de)
[11] *ALS, LBNL*, Berkeley, CA, USA (sujatagoswami@lbl.gov)
[12] *Fermilab*, Batavia, IL, USA (ntran@fnal.gov, bhawks@fnal.gov)
[13] *Discovery Partners Institute, UIUC*, Chicago, IL, USA (mcolom4@illinois.edu)
[14] *SDSC, UC San Diego*, San Diego, CA, USA (christine@sdsc.edu)
[15] *Prince Sattam bin Abdulaziz University*, Saudi Arabia (a.alsudais@psau.edu.sa)
[16] *Cirrus AI*, Johannesburg, South Africa (gregg.barrett@cirrusai.net)
[17] *Duke University*, Durham, NC, USA (tianhao.li@duke.edu)
[18] *Harvard University*, Cambridge, MA, USA (knmorehouse@gmail.com)
[20] *Microsoft*, Vancouver, BC, Canada (kartikmathur@microsoft.com)
[21] *Independent Researcher*, Tampa, FL, USA (victorjunlu@gmail.com)
[22] *Office of the CTO, Dell Technologies*, Santa Clara, CA, USA (Singh.Tejinder@Dell.com)
[23] *Cornell Tech, Cornell University*, New York, NY, USA (kzm6@cornell.edu)
[24] *Google*, Mountain View, CA, USA (kongtao@google.com)
[25] *Independent Researcher*, San Jose, CA, USA (sasidhar.kunapuli@gmail.com)
[26] *University of Padua*, Padua, Italy (gavinmichael.farrell@phd.unipd.it)
[27] *St. Jude Children's Research Hospital*, Memphis, TN (Renato.Umeton@stjude.org)

***Abstract*—Benchmarks are one cornerstone of modern machine learning practice, providing standardized evaluations that enable reproducibility, comparison, and scientific progress. However, AI benchmarks are becoming increasingly complex, requiring special care, including AI focused dynamic workflows. This is evident by the rapid evolution of AI models in architecture, scale, and capability; the evolution of datasets; and deployment contexts continuously change, creating a moving target for evaluation. Large language models in particular are known for their memorization of static benchmarks, which causes a drastic difference between benchmark results and real-world performance. Beyond the accepted static benchmarks we know from the traditional computing community, we need to develop and evolve continuous adaptive benchmarking frameworks, as scientific assessment is increasingly misaligned with real-world deployment risks. This requires the develop-**

ment of skills and education focused on benchmarks in the scientific community: *AI Benchmark Carpentry.*

Drawing on our experience from MLCommons, educational initiatives, and government programs such as the DOE's Trillion Parameter Consortium, we identify key barriers that hinder the broader adoption, utility, and evolution of benchmarking in AI. These include substantial resource demands, limited access to specialized hardware, lack of expertise in benchmark design, and uncertainty among practitioners about how to relate benchmark results to their own application domains. Moreover, current benchmarks often emphasize peak performance on leadership-class hardware, offering limited guidance for more diverse, real-world deployment scenarios. This may include applications to smaller compute resources, but also to larger systems such as LLMs deployed by commercial entities.

We argue that benchmarking itself must become dynamic in order to incorporate evolving models, updated data, and heterogeneous computational platforms while maintaining transparency, reproducibility, and interpretability. Democratizing this process requires not only technical innovation, but also systematic educational efforts as part of AI benchmark carpentry offerings, spanning undergraduate to professional levels, in order to develop sustained expertise in benchmark design and use. Finally, benchmarks should be framed and used to support application-relevant comparisons, enabling both developers and users to make informed, context-sensitive decisions. Advancing dynamic and inclusive benchmarking practices will be essential to ensure that evaluation keeps pace with the evolving AI landscape and supports responsible, reproducible, and accessible AI deployment. Furthermore, we believe that it is timely to provide a solid foundation for designing, using, and evolving benchmarks through community efforts that allows us to enable the concept of *AI benchmark carpentry.*

***Index Terms*—benchmark, AI benchmark, AI benchmark carpentry, AI benchmark democratization, MLCommons**

I. Introduction

Recently, the availability of graphics processing units (GPUs) and the rapid progress in artificial intelligence (AI) – especially in the area of deep learning – have brought a revolution to the scientific community. However, the use of these technologies is still in its infancy due to several factors. First, many application scientists are unsure how to leverage these newly available tools and instruments. Second, it remains unclear what level of effort is required to integrate them into their own research. Third, the specific demands these technologies place on infrastructure to be useful for a given scientific problem are not yet well understood.

Some of these challenges can be addressed by providing meaningful benchmarks to the scientific community, which can help researchers assess the usefulness and scalability of AI methods for their own applications. Therefore, it is beneficial to formalize the development of standardized AI benchmarks—not by a few individuals, but by the broader community. Such benchmarks can serve as a critical foundation for the scientific community, enabling rigorous evaluation, comparison, and reproducibility of new models and techniques.

However, as AI systems have become more sophisticated, incorporating complex and dynamic workflows, the traditional static approach to defining benchmarks has proven to be a significant limitation. In addition, to conventional benchmarks that capture key concepts familiar to scientists, we must also account for the continuous evolution of AI models and architectures, the changing nature of datasets, and the diversity of deployment contexts. These factors create a moving target for evaluation, risking a growing misalignment between benchmark results and the actual performance of AI systems in real-world scenarios.

Drawing on insights from our work with MLCommons, educational initiatives, and government-led projects such as the U.S. Department of Energy's Trillion Parameter Consortium [1, 2], we identify a set of fundamental barriers that impede the broader utility and adoption of AI benchmarking. Beyond the substantial resource demands and limited access to specialized, leadership-class hardware, there exists a pervasive lack of expertise in benchmark design and a growing uncertainty among practitioners regarding how to relate these performance metrics to their specific application domains. Current benchmarks—by often prioritizing peak performance on elite hardware—offer insufficient guidance for the diverse range of computational platforms encountered in practice, from smaller-scale devices to large, pre-deployed commercial language models.

This paper argues that the practice of AI benchmarking itself must become dynamic and adaptable to keep pace with the rapidly evolving AI landscape. To achieve this, benchmarks must be designed to transparently incorporate evolving models, updated datasets, and heterogeneous computational platforms, while upholding the core principles of transparency, reproducibility, and interoperability. We propose that two complementary strategies can advance this goal: first, democratizing the creation of AI benchmarks and expanding the community contributing to them; and second, establishing a robust foundation for the technical execution and innovation of benchmarks through coordinated educational efforts. Together, these approaches will foster sustained expertise spanning from undergraduate education to professional practice.

We believe it is both timely and necessary to establish a solid foundation for the design, use, and evolution of benchmarks through collaborative community efforts—thereby enabling what we call AI benchmark carpentry. This paper summarizes the collective perspectives developed through this process within the MLCommons Science & HPC Working Group.

The paper is organized as follows. In Section II, we introduce some essential definitions that we use throughout this paper. Section III introduces a formal specification for AI benchmarks. In Section IV, we summarize briefly

some existing AI benchmark efforts. In Section V, we outline how to share benchmarks. In Section VI, we define activities to be conducted as part of the educational efforts. In Section VII, we identify what we need to do to conduct democratization efforts. Lastly, we conclude in Section VIII.

Additionally, we list acronyms and abbreviations used in this paper in the Appendix A. Contributions of the authors are summarized in the Appendix B.

II. Definitions

In this section, we introduce some of the definitions and terminology used throughout this work in order to work towards a formal definition of AI benchmarks.

A. What is Benchmarking?

In computing and scientific software evaluation, benchmarking is the process of comparing metrics for computer programs, models, or systems in order to assess their relative performance, typically with respect to a baseline. While early benchmarks focused largely on hardware throughput (e.g., the time required to complete a fixed computational task), modern benchmarks increasingly evaluate software, algorithms, and integrated systems. Three dimensions now structure most benchmarking efforts: 1) runtime—the amount of time a system requires to complete a set task; 2) accuracy—the comparative quality or correctness of outcomes for the same task; and 3) efficiency—the ratio between used computational resources and quality of outcomes.

The goals of benchmarking include identifying performance gaps, establishing baseline expectations, driving innovation, and supporting continuous improvement over both short- and long-term horizons. Benchmarking has been extensively used in computer engineering and science—across both industry and academia—to measure the performance of computing equipment and the applications running on such systems.

In addition to the classical primary outcome metrics (runtime, accuracy, efficiency), today's benchmarks evaluate secondary qualities that are of high importance to the real-world deployment of systems. These include robustness and reliability (stability with respect to distribution shifts and noise, generalization), usability and accessibility (ease of integration with other systems, error transparency, ease of setup), and reproducibility (stability of the results and consistent behavior across versions, seeds, or environments).

B. Lessons Learned from Traditional HPC Benchmarking

Traditional high-performance computing (HPC) benchmarking includes:

1) *synthetic benchmarks* that simulate characteristic community workloads, as exemplified by the TOP500 and Green500 benchmarks;
2) *application benchmarks* that represent real-world applications to measure end-to-end performance, such as SPEC HPC; and
3) *scientific application benchmarks* that emphasize the accuracy of computational methods in solving domain-specific scientific problems.

(For a more detailed discussion, see Section IV-A)

Important design and applicability criteria for benchmarks include relevance and representativeness for the field, fairness, repeatability, cost-effectiveness, scalability, and transparency [3]. One caveat is that vendors may optimize hardware specifically for these benchmarks, potentially neglecting new real-world problems and emerging challenges not captured by traditional benchmark suites.

Therefore, it is essential to provide a diverse set of benchmarks so that different communities can evaluate and interpret results in terms of the performance metrics most relevant to their specific needs.

HPC benchmarking has traditionally focused on supercomputing performance comparisons, targeting compute performance [4, 5], as well as memory, communication, and storage performance [6, 7]. With the resurgence of AI and machine learning—including deep learning—it is now appropriate to explore additional lessons for benchmarking drawn from these domains.

HPC benchmarks are often executed under controlled conditions, such as those maintained by system administrators, to ensure exclusive access to hardware and eliminate interference from other users or applications. This approach allows for measurement of the best achievable performance and is frequently used to guide system procurement decisions. However, such conditions do not reflect the shared nature of most computing environments, which often include factors such as queue wait times and concurrent multi-user workloads sharing hardware resources.

C. What is Democratization?

We believe it is vital not only to allow experts and power users to participate in benchmarking efforts but also to lower barriers to entry — making powerful benchmarks, tools, knowledge, and infrastructure available to everyone, not just those with specialized resources or expertise. For benchmarking, this implies in particular to improve the following:-

a. **Accessibility:** Making benchmarks easier to use, enforcing open-source licensing.
b. **Open participation:** Encouraging community contributions through open-source development (e.g, on GitHub; shared repositories with transparent governance).
c. **Knowledge sharing:** Providing tutorials, documentation, and educational resources so that non-experts can effectively use and modify the benchmarks.
d. **Affordability:** Reducing cost barriers not only by introducing open source benchmarks, but also by allowing benchmarks to be offered at various scales and not only for leadership-class computing resources.

1) AI Software Democratization: One of the major success stories in the field of artificial intelligence is the emergence of AI-specific software libraries such as TensorFlow, PyTorch, and Jupyter Notebooks. These tools have democratized machine learning and data science by making advanced computational capabilities accessible to students, researchers, and small organizations that previously lacked the resources to develop such tools from scratch.

2) AI Hardware Democratization: One must recognize that a significant amount of progress in AI research is conducted on campus computers that are much smaller than hyperscale AI machines or leadership-class government systems. Furthermore, many scientists have begun to use *desktop* computers equipped with high-powered graphics cards. Hence, it is important to have meaningful AI benchmarks available that allow for comparisons across different scales.

D. What is Software Carpentry?

To set the stage for why we need AI benchmark carpentry, we need to first look at how the term has been introduced and is now commonly associated with software carpentry. After a more detailed analysis of software carpentry, we define the term AI benchmark carpentry.

Software Carpentry [8] was initially conceived to teach researchers in scientific fields fundamental computational and software development skills, analogous to a hammer or level in a tool belt. Thus, non-computer scientists would be able to improve the use and development of the software they need to conduct their own research while benefiting from targeted, short educational tutorials.

Today, a global community effort has sprung up since 1998 [9] that provides a number of training materials and sessions to the scientific community to we can leverage in some extend. Recently, additional areas beyond software, such as data carpentry. Together, these efforts includes:

- **Software Carpentry Core Efforts:** Teaches researchers foundational computing skills to enhance their productivity and efficiency in research tasks. This includes lessons in Programming with Python, Version Control with Git, The Unix Shell, Programming with R, Python, and using Git for version control.
- **Data Carpentry Efforts:** Teaches researchers skills necessary to work effectively and reproducibly with data in the context of specific domains. This includes lessons in the fiels of Astronomy, Ecology, Genomics, and Social Science with crosscutting topics such as Geospatial and Image Processing. Within those areas, are lessons such as Data Analysis and Visualization in R for Social Scientists, Foundations of Astronomical Data Science, and Introduction to the Command Line for Genomics [10].
- **Other Carpentry Efforts:** Library Carpentry provides lessons for information scientists, data stewards, and roles in library science, reusing some of the Software Carpentry topics adapted in a curation context. Additional lessons available include High-Performance Computing (HPC Carpentry) [11, 12].

From this list, we see that benchmark carpentry is missing.

E. What is Benchmark Carpentry?

Based on our observations in the educational and scientific communities [13], we find that similar efforts are needed to focus on benchmarking. This is more important as AI applications consume enormous resources, and properly scaling and using them requires a much deeper understanding of their time and space requirements. The hope is that, from similar benchmarks, not only can the scientist learn lessons about their own applications, but, if needed, their own benchmarks can be developed to estimate costs and effort more precisely. In addition, reproducible, portable benchmarks enable the selection and comparison of suitable hardware for the effort.

In general, we distinguish between hardware, software, and application components that significantly impact benchmarks.

On the hardware side, we deal with compute-oriented components such as CPUs, GPUs, and/or AI/neural accelerators (NPUs). Benchmarking them in the traditional way includes processing speed, core utilization, and instruction efficiency of a computer's central processing unit, data movement between xPU and main memory, to name a few. However, for AI, we also need performance in parallel computation, as well as AI workloads derived from AI kernels and applications.

As many AI applications require a large amount of *data* to be moved between memory, disks, CPU, and GPU memory, evaluating bandwidth, latency, and throughput is critical to understanding their impact on system performance. Hence, estimating and measuring the impact of, for example, assessing read/write speeds, IOPS, and access latency to identify bottlenecks in data storage systems is important.

Related to this is the *Network performance* metric, which measures bandwidth, latency, and packet loss to ensure efficient data transfer across systems, especially when parallel processing is used to address the scale required for good performance.

Benchmark carpentry should also teach

System Profiling and Monitoring principles and tools so as to measure real-time system metrics. *Interpreting Results, Analyzing Bottlenecks*, and *Optimizing Performance* are essential skills to identify limitations and improve overall performance through iterative strategies.

Benchmark Design and Reproducibility are similarly essential to allow comparative analyses among heterogeneous and also decentral benchmark runs. This includes fair, repeatable benchmarks that reflect real-world workloads and enable comparative analysis of the different components involved.

III. Towards a formal specification for AI benchmarks

As part of the MLCommons Science Working group meetings, we have identified that ingredients of ML benchmarks include:

1) Datasets (such as images, application specific scientific data, time series)
2) Tasks to be performed
3) Methods to perform these tasks (such as machine learning models, language models)
4) Metrics (runtime; accuracy; efficiency computed from the resources required for executing the task, such as space, memory usage, energy efficiency, power draw)
5) ML oriented performance impacts such as Latency impacted by the time per inference, Throughput for the inferences per second, and training time to reach target accuracy.
6) Replication which includes the ability to replicate the experiment while at the same time being able in a structured fashion to compare the results.

A. Formalization

To formalize the specification of a benchmark we introduce the following notation

$$B = (I, D, T \text{ or } W, M, C, R, V)$$

$$\begin{array}{ll} B & = \text{Benchmark} \\ I & = \text{Infrastructure} \\ D & = \text{Dataset} \\ T, W & = \text{Scientific Task or Workflow} \\ M & = \text{Metrics} \\ C & = \text{Constraint} \\ R & = \text{Results} \\ V & = \text{Version or Timestamp} \end{array}$$

Further we define the task to be executed as an application applied to a set of parameters.

$$T = (A, P)$$

$$\begin{array}{ll} A & = \text{Application} \\ P & = \text{Parameters} \end{array}$$

Alternative to a task, a workflow W can be used, if it contains multiple tasks that need to be conducted to achieve the scientific task (see Section III-D).

Each of B, I, D, T, M, R, A can have constraints C_c, where

$$c \in \{B, I, D, T, M, R, A\}$$

In case of static benchmarks, many of the parameters may be fixed. However, when defining dynamic benchmarks, we define a metric that is to be minimized while allowing a predefined set of parameters of the benchmark to be variable. Let $B_i(M)$ denote a benchmark with a fixed metric M and variations in I, D, T, C, R specified by i. We try to identify the minimum

$$min\{B_i(..., M, ...)(S_j) \mid \forall_j M(S_j)\}$$

where $M(S_j)$ is the value of the solution for the metric and S_j identifies a solution parameter set for the given metric. Please note that due to the statistical nature of the AI algorithms used in the benchmark, multiple solutions exist. However, we are not suggesting to conduct an exhaustive search of all possible solutions.

Let us assume M denotes the scientific accuracy of the benchmark; then, we look for the best scientific solution. Frequently, other restrictions are applied to the benchmark to make it tractable. While it is common to restrict the dataset, variation of the tested algorithm (the function we minimize) is often desired, since the scientific community is often not only interested in comparing hardware, but also in finding the best algorithmic solution. Such a solution can then be further studied with respect to efficiency or cost metrics.

Next, we briefly describe each of the parts that comprise a benchmark in more detail.

B. Infrastructure

Infrastructure refers to the computational and software environment required to execute the scientific task.

This includes computational hardware, software libraries, operating systems, and cloud platforms, but also power related infrastructure to operate the resources. In many cases some of these parameters are targeted by the benchmark for comparison (e.g., different types of GPUs). As a guiding principle, an attempt should be made for each single benchmark to be clearly described with as many infrastructure parameters as possible. This will foster a clear description, reproducibility, and comparability of the benchmark.

Clearly defined infrastructure will help with (a) reproducibility, as it ensures results can be reproduced across different environments, (b) fairness, as it identifies clearly the differences between different hardware and software used, (c) scalability as through comparison we can identify various scalability issues and properties, (d) efficiency, as we can assess resource use in regards to common metrics such as time, space, energy, and cost.

C. Dataset

A dataset or multiple datasets provide the input data for the scientific task to be performed. Datasets in benchmarking need to be stratified into training data (used to develop a machine learning model by direct interaction with the data), validation data (used to develop a machine learning model by indirect interaction, i.e., hyperparameter tuning), and test data (used to evaluate machine learning model performance after training). If the benchmark is concerned with hardware performance, not training any machine learning model, only a test dataset might be needed. In

many cases, it is important to provide different sizes of data sets to enable (a) a small set for fast development of the approach, and (b) a larger set that fosters scientific accuracy with longer run-times. Intermediary sizes are also sometimes needed to adapt to available resource constraints to compare them on different scales. Data should always be sufficiently described through metadata or documentation so their context within the scientific application can be determined. Together, these facilitate the establishment of (a) a ground truth that serves as the basis for evaluating scientific accuracy (b) a relevant and representative example that is influential for the scientific application, and (c) the identification of bias for data-driven applications.

We distinguish two different data sets: static and dynamic. If behavior can be tested statically, this is to be preferred; introduction of hyperparameters into a testing setup results in combinatorial explosion of possibilities, making some benchmarking approaches intractable or prohibitively expensive. In such case, constraints could be posed to restrict the benchmark to the most meaningful hyperparameters. In fact, doing this as part of the workflow could be an integral part of the benchmark. For instance, a standard runtime test of a given compute task on different GPUs does not require dynamic datasets, as it is not expected that the results will change over time; the hardware parameters are fully specified. Recent efforts have shown that, in some cases, we need to consider live data ingestion into benchmarks, for example, in earth science or health care applications, to support real-time predictions. We term such datasets *living datasets*, which are continuously updated with new data, edge cases, or corrections. Such living data sets are a special case of dynamic datasets. Such living datasets could be real-time data, but they could also be simulated using a static dataset while ingesting the data over time. While modifying the dataset the benchmark could evolve over time as the data available may be growing or become more accurate, supporting the need to identify the most accurate solution.

Living datasets allow us to maintain the relevance of a benchmarking task over time while simultaneously reacting to changes in the benchmarked systems.

It can also be used to adapt the benchmark to issues like over- and underfitting.

One additional aspect is that it can be useful to simulate such datasets and observe the changes of the benchmark when such data sets are utilized. Activities such as developing digital twins promote such approaches.

D. Scientific Task

The scientific task identifies the core challenge being evaluated while precisely identifying the purpose of the evaluated components. Typical tasks include classification, translation, reasoning, time series prediction, and planning. Through its precise definition, it sets the scope of the benchmark and introduces the community to the task to be executed and/or measured.

In more complex situations, the task itself may be a scientific workflow comprised of interacting components. In that case we may use a graph specification of the scientific task that uses subtasks that interact through edges indicating data flows and temporal executions. In that case we can use W instead of T as the specification of a workflow with properly augmented edges. Each task could have its own benchmark.

Formally, $W = (T, E)$, where T represents the collection of all tasks

$$T = \{t_1, t_2, t_3, \ldots, t_n\}$$

where n is the total number of tasks, and E indicates the dependencies between the tasks.

$$E = \{(t_i, t_j) \mid t_i, t_j \in T, t_i \neq t_j\}$$

where $(t_i, t_j) = (t_j, t_i)$.

The introduction of Workflows into the formal definition is also motivated by the recent introduction of *Agentic AI frameworks* to support automation and benchmarking of it.

E. Metrics

Metrics are quantitative measures used to assess the relative performance of the tested system in completing the scientific task. It has been shown in much previous work that the selection of the metric is the most crucial part of the benchmarking process.

The choice of metric determines many other aspects of the benchmarking purpose. For instance, by choosing runtime (e.g., wall clock time) as the main metric, it is strongly implied that the benchmark's main purpose is to find the fastest hardware or algorithmic implementation. By choosing an accuracy metric (e.g., F1 score), it is instead implied that the predictive performance (e.g., in classification tasks) is the target of the benchmark. Complex metrics can visualize trade-offs between the primitive metrics; for instance, a benchmark for the efficiency of a classification algorithm can weight its F1 score against the runtime (per sample inference speed), model size (in parameters), and energy requirements.

Implemented in this way, metrics can be used to establish a ranking of the benchmarked components, given they were measured in similar circumstances and under similar constraints.

F. Constraints

In many cases, it is necessary to constrain the benchmark to make the comparison tractable. This may include limits to training, inference, model size, or the amount of data used. Introducing constraints can (a) improve fairness while executing the benchmark (b) address operational real-world limitations, and (c) simplify the experimental

setup. Constraints can be applied to any component of the benchmark, e.g., C_I, C_D, etc.

G. Results

A benchmark must produce clear easy to comprehend results to allow evaluation of the task performed and to perform unambiguous performance evaluation. As described above, a major determinant of the informativeness of a benchmark is the choice of metrics. Performance can be evaluated on main metrics (e.g., accuracy or runtime), but often also includes a grid search of various methods, models, and hyperparameters. To simplify comparison, metric dashboards with charts and tables, as well as error analysis, are recommended. This allows (a) analysis of progress over time, (b) informing stakeholders about model capabilities, (c) identifying limitations of the tested methods, and (d) establishing a potential leader board for selecting suitable candidates that may be applicable to similar scientific tasks.

IV. Review of Benchmark Related to this Effort

This section provides an overview of key benchmarking efforts that motivated our paper. We start with HPC benchmarks and also address MLCommons benchmark efforts.

A. HPC Benchmarking

HPC benchmarking has a great impact on the activities that we report here and we can learn a lot from these efforts. Some of the most known efforts are TOP500 and Green500.

1) TOP500: The list of world's largest supercomputers has been released biannually for nearly 4 decades now and thus offers a number of important lessons in designing sustainable benchmarks. At the heart of the TOP500 scoring procedure, which yields a ranked list of 500 supercomputing installations, is the LINPACK benchmark [14], which bears the name of the namesake software library [15] for solving systems of linear equations. This linear solver package was designed in the 1970s and implemented in FORTRAN. The user guide for the library was published in 1979 and included a list of only 24 computers [15]. The following decades brought in various aspects of scaling into the software, the list sizes, and the machines submitted for inclusion in the ranking as well as data and reporting information.

2) Green500: Power and energy play a dominant role in the modern world of high-performance and distributed computing, with multi-megawatt data centers and computing facilities abound in many locations across the globe. The issues of excessive power draw and energy consumption data in the mid-2000s [16, 17] culminated in a special working group of cross-industry members [18, 19], combining the TOP500 ranking with the available power draw information from the supercomputers to yield the ranking called Green500 [20]. Since then, it is published alongside the TOP500 ranking and continues to underscore the importance of efficient energy use at large HPC installations.

3) HPC innovation: Besides the recognition of development of tools and software to facilitate the use of HPC systems and foster democratization, power consumption monitoring has been integrated at the various levels of HPC facilities, from the processing and networking elements to the data center level infrastructure. Also, by utilizing different floating-point precisions [21] the applications improve their efficiency and benefit from a great impact on the system performance due to direct targeting of the specific architectural designs.

The creation of leaderboards has led to a better understanding of the overall HPC system, but insights can be limited by misalignment of algorithm scaling and leaderboard projections. To counter misalignment, benchmarks should closely resemble the scientific task to be benchmarked. In some cases, it is informative to include end-to-end performance, including data storage limitations.

B. Machine Learning Benchmarks

Benchmarking in scientific machine learning (ML) has emerged as a critical area to guide algorithm development, enable fair comparisons towards progress and innovation, and facilitate reproducibility. The development of ML benchmarks for science is especially critical because of the multi-disciplinary nature of the development, often including domain experts, computing hardware developers, and ML researchers. That, coupled with the variety of tasks and workloads, makes *high quality* benchmarking critical to making progress.

To obtain an overview how many academic benchmarks have been published in well known public domain archives, so we queried arXiv [22] and Google Scholar [23]. Note that according to Google, Google Scholar does not include all entries from arXiv, but it does include most of them. However, it also includes many more resources, so we expect a larger number from Google Scholer. As of Oct 1, 2025, we find 106 entries on arXiv when searching for the topic *"AI benchmark"*. executing equivalent queries in Google Scholar yields 2,490 entries for *"AI benchmark"*. It is evident from this that a complete survey of these papers is difficult to achieve through manual inspection. In an upcoming effort, we plan to explore how to automatically categorize these entries using LLMs while implementing an agentic AI framework for it.

The vast number and diversity of scientific tasks poses challenges to finding a well-defined, high-quality benchmark for any given task. To improve discoverability, we have cataloged in this paper all MLCommons benchmarks that have a result submission. Secondly, we have developed an ontology [24, 25] that allows users to identify suitable benchmarks.

1) MLCommons: MLCommons [26] provides one of the most comprehensive and standardized ecosystems of AI

benchmarking. It addresses training, inference, scientific computing, and domain-specific benchmarks. Most prominently, the MLPerf benchmark suite—covering datacenter, edge, mobile, and training applications—establishes industry-wide baselines for performance, accuracy, power efficiency, and quality of service across diverse model classes such as computer vision, language, recommendation, speech, and reinforcement learning. Additionally, it offers specialized evaluations including MLPerf Tiny for microcontroller-class devices, MLPerf Storage for I/O workloads, and MLPerf Science for large-scale scientific AI. Furthermore, MLCommons promotes the reproducibility through initiatives such as Croissant ML, a standardized metadata schema for datasets, and MLCube, a portable container-based model packaging standard. Additional domain-specific working groups in medical AI, multilingual speech, and responsible AI have recently expanded the targeted domains.

We have provided a comprehensive list of benchmarks in Tables I and II. The tables contain information about the benchmark name, model, task, domain, model type, metrics, hardware, and a brief note. The evaluations of the AILuminate benchmarks can be found on the MLCommons Web pages and include (a) Safety / Jailbreak Tests, (b) LLM Safety Evaluation, (c) Responsible AI / Alignment (d) LLM (Decoder) (e) Safety Rate, Toxicity Score (f) Cloud LLM APIs (g) Robustness and Alignment.

2) Ontology: To improve discoverability of suitable benchmarks for a given task, we introduce a definition and AI Benchmark ontology of scientific machine learning benchmarks, where benchmarks are classified and mapped to their scientific domain and machine learning task type in [25]. This work grew out of the Web page created at [24], [27] and provides an easy to use interactive mechanism to query the cataloged benchmarks.

New AI benchmarks are added through an open submission workflow overseen by the MLCommons Science Working Group. Each submission is evaluated against a rubric of currently six categories (Software Environment, Problem Specification, Dataset, Performance Metrics, Reference Solution, Documentation) that assigns an overall rating and potential endorsement. The scoring framework enables stakeholders, researchers, domain scientists, and hardware vendors to identify representative subsets of benchmarks that align with their specific priorities. The ontology supports adding new scientific domains, AI/ML motifs, and computing motifs.

A subset of information collected by the Web page is shown in Table III. It not only includes some elementary information about the benchmarks but also a perceived rating displayed as a radar chart. Such radar charts include ratings from 1-5, where 5 is the best rating. Ratings are identified for documentation, specification, software, metrics, dataset, and reference solution. The Web page not only includes an automatically generated report of all benchmarks in PDF format, but also a convenient online publication of the benchmarks with convenient search capabilities.

TABLE I: MLCommons Benchmarks

Benchmark Name	Model	Task	Application Domain / Use Case	Model Type / Architecture	Metrics / KPIs	Hardware	Notes / Description
MLPerf Inference: Datacenter							
deepseek-r1	DeepSeek R1 (671B params)	Reasoning / Code Generation	Knowledge & Reasoning, Complex Problem Solving, Step-by-Step Planning	Large Language Model (LLM), Reasoning LLM, High context/output length (up to 20K tokens)	Accuracy: Exact Match, Code Evaluation; Latency: TTFT (Time to First Token), TPOT (Time Per Output Token)	Data Center GPUs (NVIDIA H100/H200) with massive VRAM, optimized for 671B parameters.	The model's large output length emphasizes its use in complex reasoning chains. Requires powerful systems (e.g., multiple H100 GPUs).
dlrm-v2-99	DLRM-v2	Recommendation	Personalized product/content recommendation (e.g., e-commerce, social media feeds)	Deep Learning Recommendation Model (DLRM), Sparse/Dense Architecture	Throughput: Queries Per Second (QPS); Latency: 99th Percentile Latency	Data Center CPUs and GPUs (NVIDIA B200/GB200/B300), prioritizing high I/O and memory bandwidth for massive embedding tables.	Tests high-throughput, low-latency deployment for online services with a 99% latency constraint.
dlrm-v2-99.9	DLRM-v2	Recommendation	Personalized product/content recommendation (e-commerce, social media feeds)	Deep Learning Recommendation Model (DLRM), Sparse/Dense Architecture	Throughput: Queries Per Second (QPS); Latency: 99.9th Percentile Latency	Data Center CPUs and GPUs (NVIDIA H200), often using higher precision to ensure quality target is met.	Tests high-throughput, very low-latency deployment for critical online services with a strict 99.9% latency constraint.
llama2-70b-99	Llama 2 (70B params)	Large Language Model (LLM) Inference	General text generation, chat, summarization, and understanding	LLM, Transformer-based	Throughput: Tokens Per Second (TPS); Latency: TTFT, TPOT (99th Percentile)	Data Center GPUs (e.g., AMD MI300X/MI325X, NVIDIA B200/GB200/H100/H200/L40S, MS-Intel Arc Pro B60) in multi-GPU configurations, focused on high throughput and low latency.	Represents a larger LLM workload, measuring performance under a 99% latency constraint.
llama2-70b-99.9	Llama 2 (70B params)	Large Language Model (LLM) Inference	General text generation, chat, summarization, and understanding	LLM, Transformer-based	Throughput: Tokens Per Second (TPS); Latency: TTFT, TPOT (99.9th Percentile)	Data Center GPUs (AMD MI300X/MI325X, NVIDIA B200/GB200/H100/H200/L40S, MS-Intel Arc Pro B60), often testing the limits of precision vs. speed trade-offs.	Represents a larger LLM workload, measuring performance under a stricter 99.9% latency constraint.
llama3.1-8b-datacenter	Llama 3.1 (8B params)	Summarization / Text Generation	Low-cost, high-volume LLM services, interactive code assistants	LLM, Transformer-based	Accuracy: ROUGE metrics (1, 2, L); Latency: TTFT $\leq$2s, TPOT $\leq$100ms (Server)	Single-node systems or smaller GPU clusters, used to lower the entry barrier for the MLPerf Training suite.	Benchmarks a smaller LLM for efficient deployment in both Data Center and Edge scenarios.
llama3.1-405b	Llama 3.1 (405B params)	Large Language Model (LLM) Inference	Generative AI, high-capability models	LLM, Transformer-based	Throughput: Output Tokens per second; Latency: TTFT, TPOT	Large-scale AI Clusters and Supercomputers (requires hundreds of GPUs (NVIDIA B200/GB200/GB300/H100/H200) with high-speed interconnects).	One of the largest LLMs in the suite, demonstrating the need for advanced parallelism (tensor, pipeline) on high-end systems (e.g., NVIDIA H200).
mixtral-8x7b	Mixtral (46.7B total params)	Large Language Model (LLM) Inference	generative AI, multilingual tasks	Mixture-of-Experts (MoE) LLM (activates $\approx$13B params per token)	Throughput: Tokens Per Second (TPS); Latency	Data Center GPUs (AMD MI300X/MI325X, NVIDIA H200/RTX PRO 6000), optimizing MoE architecture for low active compute per token.	Showcases the efficiency of MoE architecture, offering high quality with lower active compute cost than dense models.
retinanet	Retinanet-ResNext50	Object Detection	Identifying and localizing objects in images	Object Detection Model, often with ResNext backbone and FPN	Accuracy: mAP (mean Average Precision); Throughput: Samples Per Second	Data Center and Edge GPUs (NVIDIA GeForce RTX 4090/H200/L4-PCIe/L40S), measuring both throughput and latency under a 100ms constraint.	A standard computer vision benchmark using the OpenImages dataset.
rgat	Relational Graph Attention Network	Node Classification	Graph data analysis, social network processing, knowledge graphs	Graph Neural Network (GNN), Graph Attention Network (GAT) variant	Accuracy (on node classification); Throughput: Samples Per Second	Data Center GPUs (NVIDIA B200), specifically testing performance on irregular, graph-structured data.	Addresses graph-structured data and multi-relational graphs, testing system efficiency for complex graph workloads.
stable-diffusion-xl	Stable Diffusion XL (SDXL)	Text-to-Image Generation	Generative AI for creating high-quality images from text prompts	Diffusion Model (Latent Diffusion)	Throughput: Images Per Second; Latency	Data Center and Professional GPUs (AMD MI325X,NVIDIA B200/H100/H200/L4-PCI/L40S/NVIDIA RTX PRO 6000), focusing on the speed of image generation (samples/second).	Represents the Text-to-Image Generative AI domain, measuring the speed of image synthesis.
whisper	Whisper-Large-V3	Automatic Speech Recognition (ASR)	Converting spoken audio to text	Encoder-Decoder Transformer, Speech-to-Text Model	Accuracy: WER (Word Error Rate), Word Accuracy (Acc); Latency	Data Center GPUs (NVIDIA B200/GB200/GeForce RTX 4090/H100/H200/L4-PCIe/L40S), measuring performance on a complex sequence-to-sequence model for speech.	An ASR benchmark on multilingual audio, measuring both encoder (audio feature) and decoder (token generation) performance.
MLPerf HPC							
CosmoFlow	CosmoFlow 3D CNN	Regression	Astrophysics, Cosmology (predicting properties of the universe from simulation data)	3D Convolutional Neural Network (3D CNN)	Time to Quality (TTQ) (e.g., Time to reach validation MAE $\leq$ 0.124)	Supercomputers & Large HPC Clusters (e.g., Fugaku, Perlmutter). Stresses distributed training, 3D data handling, and fast data I/O for massive volumetric datasets ($\approx$ 5 TB) GPUs used for running this benchmark: NVIDIA A100/V100.	Uses massive 3D volumetric data ($\approx$ 5.1 TB). Stresses memory bandwidth and interconnect.
DeepCAM	DeepCAM Encoder-Decoder	Semantic Segmentation	Climate Science, Extreme Weather Prediction (identifying atmospheric rivers, tropical cyclones)	Convolutional Encoder-Decoder (e.g., U-Net or DeepLab-like)	Time to Quality (TTQ) (e.g., Time to reach validation IoU $\geq$ 0.82)	Supercomputers & Large HPC Clusters. Stresses large-scale image processing, high-dimensional data (many channels), and efficient communication on systems with thousands of GPUs (A100/P100/V100).	Trained on massive, high-resolution 2D image data ($\approx$ 8.8 TB). Stresses I/O and communication efficiency.

Continued on next page

TABLE I: MLCommons Benchmarks (Cont.)

Benchmark Name	Model	Task	Application Domain / Use Case	Model Type / Architecture	Metrics / KPIs	Hardware	Notes / Description
OpenCatalyst	DimeNet++	Regression	Computational Chemistry, Materials Science (discovering new catalysts for energy storage)	Graph Neural Network (GNN)	Time to Quality (TTQ) (Time to reach target energy/force prediction error)	Supercomputers & Large HPC Clusters. Stresses performance on graph-structured data (atomic systems) and complex GNN operations that require high GPU utilization. GPUs used for running this benchmark: NVIDIA A100/P100/V100.	Models atoms and bonds as a graph structure. Benchmarks complex, irregular GNN workloads at scale.
MLPerf Training							
BERT (Bidirectional Encoder Representations from Transformers)	NLP - Question Answering	General NLP, Text Understanding	Transformer (Encoder)	Time to Quality (TTQ) (F1 Score on SQuAD)	Data Center GPUs, Accelerators	CPU, Single GPU (e.g., NVIDIA A100/H100), or moderate clusters.	A foundational benchmark for Natural Language Processing tasks.
DLRM-dcnv2 (Deep Learning Recommendation Model - DCNv2)	Recommendation Systems	E-commerce, Content Streaming, Personalized Ads	Deep Learning Recommendation Model w/ DCNv2	Time to Quality (TTQ) (AUC on Criteo 4TB)	Data Center GPUs, Specialized Accelerators	Large-scale GPU clusters with high-speed interconnects (e.g., InfiniBand) for distributed training. This benchmark was running on GPUs: NVIDIA B300/B200/GB200/H200/H100/H200.	Stresses memory bandwidth and communication for massive embedding tables.
llama2-70b-lora	LLM Fine-Tuning	Customizing LLMs for specific enterprise tasks	Transformer with LoRA	Time to Quality (TTQ) (ROUGE Score)	Multi-GPU servers, Mid-size GPU clusters	High-end Multi-GPU servers or small clusters (e.g., systems with AMD MI300X/MI325X/MI350X/MI355X, NVIDIA B200/B300/H100/H200).	Measures the efficiency of Low-Rank Adaptation (LoRA) on a ≈ 70B parameter model.
llama3.1-405b	LLM Pretraining	Generative AI, Foundational Model Development	Transformer-based LLM (≈ 405B params)	Time to Quality (TTQ) (Log Perplexity)	Large-scale, Multi-node GPU clusters	Single Node or small GPU systems (e.g., a few GPUs per node) to keep the benchmark accessible.Benchmark running on GPUs: NVIDIA B200/B300/H200.	The largest, most compute-intensive benchmark for pretraining state-of-the-art LLMs.
RetinaNet	Object Detection	Autonomous Vehicles, Surveillance, Image Analysis	One-stage Object Detector (ResNet, FPN)	Time to Quality (TTQ) (mAP on COCO)	Data Center GPUs, Cloud Instances	Single or multi-GPU systems (NVIDIA B200/H200/RTX Pro 6000), often used in both Datacenter and Edge devices for inference.	Measures performance for a core computer vision task: localizing and classifying objects.
RGAT (Relational Graph Attention Network)	GNN - Node Classification	Drug Discovery, Social Network Analysis, Fraud Detection	Relational Graph Attention Network (R-GAT)	Time to Quality (TTQ) (Accuracy on IGBH)	Systems optimized for high-bandwidth interconnects	GPU-based systems (NVIDIA B200/B300/H100), optimized for workloads with complex, sparse data structures like graphs.	Focuses on the irregular memory access and communication patterns of Graph Neural Networks.
Flux1 (stable-diffusion)	Text-to-Image Generation	Generative AI, Digital Art, Content Creation	Latent Diffusion Model (U-Net, Transformer)	Time to Quality (TTQ) (FID and CLIP Scores)	Multi-GPU servers, Cloud Instances	High-performance Single or Multi-GPU systems (especially for fast inference or training). This benchmark was running on: NVIDIA B200/GB200/GB300.	Benchmarks the training of a major generative model in the AI industry.
MLPerf Inference: Edge							
3D U-Net (99%)	3D U-Net	Medical Image Segmentation	Healthcare, Volumetric Imaging (e.g., MRI/CT)	3D Convolutional Encoder-Decoder CNN	Accuracy (Dice Score), Latency, Throughput (QPS)	Data Center GPUs (e.g., NVIDIA A100/H100), high-performance computing (HPC) systems, specialized accelerators.	99% of reference accuracy target. Typically runs in Offline scenario for batch processing of medical scans.
3D U-Net (99.9%)	3D U-Net	Medical Image Segmentation	Healthcare, High-Fidelity Imaging	3D Convolutional Encoder-Decoder CNN	Accuracy (Dice Score), Latency, Throughput (QPS)	Data Center GPUs (e.g., NVIDIA A100/H100), high-performance computing (HPC) systems, specialized accelerators.	99% of reference accuracy target. Represents a stricter quality constraint, often requiring higher-precision compute (e.g., FP16 vs. INT8).
llama3.1-8b-edge	Llama 3.1 (8B params)	Text Generation / Summarization	Edge AI, On-device LLMs, Interactive Assistants	Quantized Transformer (Decoder-only LLM)	Tokens Per Second (TPS), Latency (TTFT, TPOT), Power	Edge devices, mobile SoCs (System-on-Chips), smaller GPUs (MS-Intel Arc Pro B60), high-end CPUs.	Benchmarks a modern, smaller LLM variant optimized for performance and low-latency on resource-constrained Edge devices.
resnet	ResNet50-v1.5	Image Classification	Vision, Quality Control, Surveillance	CNN (Residual Network)	Accuracy (Top-1), Latency, Throughput (QPS)	Data Center GPUs (NVIDIA GeForce RTX 4090/RTX-2000E), Edge devices, Mobile SoCs, CPUs, specialized accelerators.	The foundational computer vision benchmark, often used as a baseline for measuring performance and efficiency across all MLPerf tiers.
retinanet	RetinaNet-ResNext50	Object Detection	Autonomous Vehicles, Advanced Security Systems	One-stage Object Detection (often with FPN)	Accuracy (mAP - mean Average Precision), Latency, Throughput (SPS)	Data Center GPUs (NVIDIA GeForce RTX 4090/4000/2000E), Edge devices, specialized detection accelerators.	Measures the system's ability to find and localize multiple objects in images. Uses the OpenImages dataset.
stable-diffusion-xl	Stable Diffusion XL (SDXL)	Text-to-Image Generation	Generative AI, Digital Content Creation	Diffusion Model (Latent Diffusion with U-Net)	Images Per Second, Latency (Time to generate an image)	Data Center GPUs (e.g., NVIDIA H100/H200, AMD MI300 series), powerful consumer-grade GPUs.	Represents the high-compute generative AI domain. Measures the speed of synthesizing high-resolution images from text prompts.
whisper	Whisper-Large-V3	Automatic Speech Recognition (ASR)	Speech-to-Text Services, Live Transcription	Encoder-Decoder Transformer	Accuracy (WER - Word Error Rate, Word Acc), Tokens Per Second	Data Center GPUs (NVIDIA GeForce RTX 4090), Edge/Client devices for real-time transcription.	A modern, high-accuracy ASR benchmark, using a Transformer architecture that handles both audio encoding and token generation.
MLPerf Inference: Mobile							

Continued on next page

TABLE I: MLCommons Benchmarks (Cont.)

Benchmark Name	Model	Task	Application Domain / Use Case	Model Type / Architecture	Metrics / KPIs	Hardware	Notes / Description
MLPerf Mobile/Edge	MobileNetV4-Conv-L	Image Classification, Object Detection	Edge/Mobile AI, low-latency on-device vision tasks.	CNN / MobileNet Family (V4)	Latency (ms), Throughput (Inferences/sec), Top-1/Top-5 Accuracy, Average Precision (AP).	Mobile SoCs, Specialized Mobile Accelerators (e.g., Apple Neural Engine, Edge TPUs, dedicated DSPs)	The largest convolutional-only variant of MobileNetV4. Optimized via Neural Architecture Search (NAS) for better latency-accuracy trade-offs on mobile and embedded hardware.
MLPerf Mobile/Edge	Mobile SSD Variants	Object Detection	Edge/Mobile AI, real-time detection on resource-constrained devices.	Single Shot Detector (SSD) + Mobile Backbone	Average Precision (AP) (e.g., COCO AP), Latency (ms), FPS.	Mobile SoCs (CPU, GPU, NPU/DSP), Edge AI Accelerators	Refers to models like SSD-MobileNet V1/V2/V3 which are standard mobile benchmarks.
MLPerf Edge	SSD-MobileNet	Object Detection (Small)	Edge/Mobile AI, detection for systems with tight latency/power budgets.	Single Shot Detector (SSD) + MobileNet Backbone	Average Precision (AP), Latency (ms).	Mobile SoCs (CPU, GPU, NPU/DSP), Edge AI Accelerators	A specific variant that is an original, primary benchmark for MLPerf Inference: Edge.
MLPerf Mobile/Edge	MobileNet V1–V4	Image Classification, Feature Extractor	Efficient Vision Models, low-power and low-latency inference.	CNN (V1: Depthwise Separable Convs, V2: Inverted Residuals, V3: Squeeze-and-Excitation, V4: UIB/-Mobile MQA)	MACs/FLOPs, Latency (ms).	Mobile SoCs (CPU, GPU, NPU/DSP, e.g., Qualcomm Snapdragon, Apple A-series), Microcontrollers (MCUs), Edge AI Accelerators (e.g., Google Edge TPU)	A progression of architectures from Google, all focused on minimal computational cost while maintaining high accuracy, crucial for all MLPerf Edge divisions.
MLPerf Mobile	MobileNet V4	Image Classification, Object Detection	Universally Efficient AI, aiming for state-of-the-art accuracy-latency trade-offs.	Hybrid (Convolutional + Attention - Mobile MQA)	Latency (ms)	Mobile SoCs (CPU, GPU, NPU/DSP, e.g., Qualcomm Snapdragon)	The latest generation, featuring the Universal Inverted Bottleneck (UIB) and Mobile MQA.
MLPerf Mobile	MOSAIC	Image Segmentation	Mobile Image Segmentation, on-device image processing.	U-Net variant with a MobileNet-style backbone.	Mean Intersection over Union (mIoU), Latency (ms).	Mobile SoCs (CPU, GPU, NPU)	A common model used for segmentation tasks in the MLPerf Mobile suite.
MLPerf Mobile	MobileDETs	Object Detection	Edge/Mobile AI, high-speed detection for mobile chips.	Model Family derived from Neural Architecture Search (NAS)	Average Precision (AP), Latency (ms).	Mobile SoCs (NPU/DSP emphasized), Edge AI Accelerators	A family of detectors specifically optimized for latency on mobile SoCs.
MLPerf TinyMobile	BERT-Tiny DistilBERT	Natural Language Processing (NLP) Tasks (e.g., Q&A)	MobileEdge NLP, faster, smaller language understanding on local devices.	Transformer Distillation Models	Latency (ms), F1 Score (SQuAD), GLUE Score.	CPUs, GPUs, Edge AI Accelerators, Mobile SoCs (optimized for low-latency)	Smaller, compressed versions of BERT achieved through knowledge distillation for resource-constrained environments.
MLPerf Mobile	Mobile-BERT	Natural Language Processing (NLP) Tasks	Edge/Mobile NLP, task-agnostic BERT for resource-limited devices.	Compressed Transformer (Bottleneck structures, Knowledge Distillation)	Latency (ms), F1 Score (SQuAD), GLUE Score.	CPUs, GPUs, Edge AI Accelerators, Mobile SoCs (optimized for low-latency)	Achieves competitive results to BERT-Base with much higher speed and smaller size.
MLPerf Mobile	EDSR F32B5	Image Super-Resolution (SR)	Image Enhancement, upscaling low-resolution images for improved quality.	Enhanced Deep Super-Resolution (EDSR) Network	Latency (ms), PSNR, SSIM.	GPUs, Custom Hardware/FPGAs, specialized ISP (Image Signal Processor) components.	A common, high-quality reference model for measuring performance on image enhancementquality tasks.
MLPerf Mobile	Stable Diffusion	Text-to-Image Generation	Generative AI, creating high-resolution images from text prompts.	Latent Diffusion Model (LDM) (U-Net, VAE, CLIP Text Encoder)	Images/Query Per Second (Throughput), Latency (Time-to-Image), FID/CLIP Scores.	High-end GPUs (e.g., NVIDIA A100/H100, RTX series), high-power Workstations and Data Center Accelerators.	A critical benchmark for measuring performance on large, complex generative workloads.
MLPerf Inference: Tiny							
MLPerf Tiny v 0.5	Keyword Spotting Model	Audio Classification	TinyML/MCU, always-on voice assistant, device wake-word detection.	Small CNN (e.g., DS-CNN) or RNN.	Latency (ms), Energy (Joules), Area Under the ROC Curve (AUC).	Microcontrollers (MCUs) (e.g., Arm Cortex-M4M7), Digital Signal Processors (DSPs), Tiny Neural Network Accelerators.	Detects a specific word (e.g., "Hey Google") from a stream of audio, running on a highly constrained power budget.
MLPerf Tiny v 0.5	Visual Wake Words (VWW) Model	Image Classification (Binary)	TinyML/MCU, low-power sensing, person detection, motion-activated cameras.	Small CNN (e.g., MobileNet V1/V2 variant).	Latency (ms), Energy (Joules), AUC.	MCUs, low-power vision processors, small-scale embedded systems.	Determines if a person is present in the image (person/not-person). Much simpler and smaller than general ImageNet classification.
MLPerf Tiny v 0.5	Image Classification Model	Image Classification (Multi-class)	TinyML/MCU, general object recognition on ultra-low-power sensors.	Very small CNN (e.g., ResNet-8 or Micro-CNN).	Latency (ms), Energy (Joules), Top-1 Accuracy (e.g., on CIFAR-10).	MCUs with limited RAM and Flash storage.	A more complex classification task than VWW, but still constrained to a very small model size.
MLPerf Tiny v 0.5	Anomaly Detection (AD) Model	Time Series Anomaly Detection	TinyML/MCU, industrial predictive maintenance, system health monitoring.	Small Autoencoder or similar lightweight model.	Latency (ms), Energy (Joules), AUC.	MCUs, industrial IoT sensors, devices monitoring vibration or sound.	Learns a baseline of normal sensor data (e.g., machine vibrations) and flags deviations as anomalies.
MLPerf Client							
MLPerf Client	Llama 2 7B Chat	Code analysis, Content generation, Creative writing, Summarization (various lengths).	General-purpose AI, Dialogue/Chatbots, Client-side LLM inference on PCs.	Transformer, Decoder-Only, Instruction-Tuned (SFT + RLHF), 7 Billion parameters.	Time-to-First Token (TTFT), Tokens/Second (Throughput).	Client GPUs (e.g., AMD Radeon, Intel Arc), Integrated NPUs (e.g., Intel Core Ultra, AMD Ryzen AI), Data Center GPUs (e.g., NVIDIA A100/H100) for server-side inference.	A foundational model in the benchmark for measuring core client-side LLM performance.
MLPerf Client	Llama 3.1 8B Instruct (8B parameters)	Generative AI workloads: Code analysis, Content generation, Creative writing, Summarization.	General-purpose AI, Instruction Following, Client-side LLM inference on PCs.	Transformer, Decoder-Only, Instruction-Tuned, 8 Billion parameters.	Time-to-First Token (TTFT) (Latency), Tokens/Second (Throughput).	Client PCs and Data Center/Cloud-based GPUs (optimized for both low-latency "Time to First Token" and high-throughput "Tokens Per Second").	An updated and highly capable open-weight model, demonstrating improved performance and alignment over Llama 2.
MLPerf Client	Phi 3.5 Mini Instruct	Reasoning (Math, Code, Logic), Long Context Query & Summarization (up to 128K tokens).	Memory/Compute Constrained Environments, Low-Latency Applications, On-device deployment (AI PCs, mobile).	Dense Decoder-Only Transformer, Instruction-Tuned, 3.8 Billion parameters.	Time-to-First Token (TTFT) (Latency), Tokens/Second (Throughput).	Client GPUs, NPUs, and potentially high-end mobile/edge processors (optimized for on-device deployment).	A highly efficient and lightweight model optimized for speed and strong reasoning despite its small size.

Continued on next page

TABLE I: MLCommons Benchmarks (Cont.)

Benchmark Name	Model	Task	Application Domain / Use Case	Model Type / Architecture	Metrics / KPIs	Hardware	Notes / Description
MLPerf Client	Phi 4 Reasoning 14B	Complex Reasoning (multi-step math, scientific, coding, planning), Generating detailed chain-of-thought traces.	Agentic applications, High-accuracy problem-solving, Applications requiring explainability.	Dense Decoder-Only Transformer, Reasoning-Focused SFT (and possible RLHF for Plus variant), 14 Billion parameters.	Time-to-First Token (TTFT) (Latency), Tokens/Second (Throughput), Accuracy on reasoning tasks.	High-performance Client PCs (Workstations) and Data Center GPUs (due to its larger size and focus on complex, token-intensive reasoning).	Included as an experimental model in the benchmark, specifically designed to emphasize logical and complex problem-solving.
MLPerf Storage							
MLPerf Storage	ResNet-50	I/O Workload for Image Classification Training	General-purpose computer vision, low-latency image processing.	Convolutional Neural Network (CNN)	Max Supported Accelerators, Aggregate Throughput (MiB/s), Accelerator Utilization ($\geq$ 90% required).	Data Center GPUs (NVIDIA A100/H100), Edge AI Accelerators, and high-end CPUs (widely used across all MLPerf divisions: Data Center, Edge, Tiny).	High IOPS Demand. Characterized by highly concurrent, random reads of many small data samples ($\approx$ 150 KB each), stressing metadata and IOPS capability.
MLPerf Storage	3D U-Net	I/O Workload for Medical Image Segmentation Training	Healthcare/Radiology, Medical Image Analysis, 3D data processing.	3D U-Net (3D CNN)	Max Supported Accelerators, Aggregate Throughput (GiB/s), Accelerator Utilization ($\geq$ 90% required).	High-end Data Center GPUs (NVIDIA A100/H100) and specialized high-throughput storage systems (MLPerf Storage benchmark focus).	High Bandwidth Demand. Characterized by concurrent random reads of very large data files ($\approx$ 140 MB each), stressing sustained data throughput.
MLPerf Storage	CosmoFlow	I/O Workload for Scientific Parameter Prediction Training	Scientific High-Performance Computing (HPC), Astrophysics.	3D Convolutional Neural Network (3D CNN)	Max Supported Accelerators, Aggregate Throughput (GiB/s), Accelerator Utilization ($\geq$ 70% required).	Supercomputers & HPC Clusters: Requires massive scale distributed training across hundreds or thousands of GPUs (e.g., utilizing NVIDIA H100s, Intel Gaudi, and specialized high-speed interconnects like InfiniBand).	CPU-Intensive Workload. Uses medium-sized samples ($\approx$ 2 MB), but the client-side processing is more CPU-heavy, leading to a slightly lower required accelerator utilization threshold.
MLPerf Automotive							
MLPerf Automotive	SSD-ResNet50	2D Object Recognition and Segmentation	ADAS / Collision Avoidance, Lane Departure	Single Shot Detector (SSD) with ResNet-50 Backbone	Latency, Throughput, mAP (Accuracy)	Edge AI Accelerators, Embedded GPUs, and Automotive System-on-Chips (SoCs).	Baseline benchmark for camera-based detection on high-res (8MP) images. Used in v0.5.
MLPerf Automotive	BEVFormer-Tiny	Camera-based 3D Object Detection	Autonomous Driving (L2+ to L4), Environmental Perception	Bird's Eye View (BEV) Transformer-based Network	Latency, Throughput, mAP (Accuracy)	High-compute Automotive SoCs, next-generation AI accelerators (specifically targeting transformer and multi-sensor fusion capabilities).	Represents state-of-the-art camera-only 3D perception. Used in MLPerf Auto v0.5.
MLPerf Automotive	DeepLabV3Plus / PointPainting	Semantic Segmentation (as a component of 3D Detection)	Lidar-Camera Sensor Fusion, 3D Perception	DeepLabV3+ (for Segmentation) + PointPillars (for 3D Detection)	Latency (p99.9 percentile), Throughput, Accuracy)	Safety-critical Automotive SoCs, purpose-built AI processors for ADAS/AV, often requiring high-reliability and low-latency performance.	DeepLabV3+ is the 2D segmentation part of the PointPainting sensor fusion pipeline. Used in MLPerf Inference v5.0 Automotive.
MLPerf Training:HPC							
MLPerf Training:HPC	CosmoFlow	Prediction of Cosmological Parameters ($\Omega_m, \sigma_8, n_s, H$)	Astrophysics, Cosmology, Scientific Simulation Parameter Prediction	3D Convolutional Neural Network (3D CNN)	Time-to-Train (Total time to reach a target quality metric), Aggregate Throughput (Models trained per unit of time in weak scaling).	Supercomputers & Large Clusters (e.g., NVIDIA Selene, Perlmutter, Fugaku), utilizing thousands of interconnected High-Performance GPUs (e.g., NVIDIA A100/H100) and high-speed parallel file systems.	Trained on 3D volumetric data (dark matter distributions) from N-body simulations. The large, volumetric data introduces significant I/O challenges and stresses high-bandwidth interconnects and storage.
MLPerf Training:HPC	DeepCAM	Semantic Segmentation of Extreme Weather Events (e.g., atmospheric rivers, tropical cyclones)	Climate Science, Weather Forecasting, Earth System Modeling	Convolutional Encoder-Decoder (U-Net variant)	Time-to-Train (Total time to reach a target quality metric), Aggregate Throughput (Models trained per unit of time in weak scaling).	Supercomputers & Large Clusters, demanding high I/O bandwidth to handle the massive 8.8 TB climate datasets and requiring excellent strong-scaling performance. This benchmark was running on NVIDIA V100/A100.	Trained on massive, high-resolution, multi-channel images (e.g., 768 $\times$ 1152 pixels with 16 channels). Features high computational intensity and large memory footprint per sample.
MLPerf Training:HPC	OpenCatalyst	Prediction of energy and forces for molecular systems (AI for materials science)	Catalyst Discovery, Computational Chemistry, Materials Science, Energy Storage	Graph Neural Network (GNN), specifically DimeNet++	Time-to-Train (Total time to reach a target quality metric), Aggregate Throughput (Models trained per unit of time in weak scaling).	Supercomputers & Large Clusters, typically emphasizing the performance of GNNs, which stress different aspects of the system, like memory access patterns and graph-specific operations. This benchmark was running on NVIDIA V100/A100.	Predicts quantum mechanical properties of catalyst systems. Stresses complex data structures (graphs) and large-scale parallel processing. Uses the massive OC20 dataset.
MLCommons Science							
MLCommons Science	Cloud Mask	image processing / segmentation	Earth Observation, Segmentation model for the pixel classification in satellite images	U-Net deep neural network	training and inference timing and scalability on the training across a number of GPUs;runtime of training and inference.	HPC Clusters & High-Performance GPUs (e.g., NVIDIA A100/V100) running distributed training frameworks like PyTorch or TensorFlow, often benchmarked for large-scale data I/O.	Focuses on identifying and isolating cloud cover in high-resolution satellite imagery for subsequent analysis.

Continued on next page

TABLE I: MLCommons Benchmarks (Cont.)

Benchmark Name	Model	Task	Application Domain / Use Case	Model Type / Architecture	Metrics / KPIs	Hardware	Notes / Description
MLCommons Science	STEMDL	A universal classifier for the space group of solid-state materials.	Scientific Machine Learning (General benchmark suite)	CNN: ResNet, VGG, DenseNet	top1 accuracy and F1 score (Macro)	HPC Systems of all sizes, used for general performance comparison across different hardware architectures and scaling tests. This benchmrak was running NVIDIA A100/V100.	The goals of this benchmark are to: (1) explore the suitability of machine learning algorithms in the advanced analysis of Convergent beam electron diffraction (CBED) and (2) produce a machine learning algorithm capable of overcoming intrinsic difficulties posed by scientific datasets.
MLCommons Science	CANDLE UNO	Cancer Drug Response Prediction	Life Sciences / Personalized Medicine	Neural Networks(MLP)	TTT, Prediction Accuracy	HPC Systems (e.g., Summit, Polaris) and Cloud Environments, stressing both compute performance and workflow management for parameter sweep tasks. This benchmark was running NVIDIA A100.	Benchmarks deep learning models for predicting the response of various cancer cell lines to different therapeutic compounds.
MLCommons Science	Earthquake	TEvolOp Earthquake Forecasting Model	Earthquake Science	Neural Networks(MLP)- recurrent neural networks and transformers	Nash Sutcliffe efficiency	HPC & Big Data Systems, requiring efficient handling of large, continuous time-series datasets and high-throughput data processing. This benchmark was running on NVIDIA V100.	Benchmarks deep learning models for predicting the response of various cancer cell lines to different therapeutic compounds.
MLCommons AlgoPerf							
AlgoPerf	Criteo 1TB	Click-Through Rate (CTR) Prediction	Large-scale Recommender Systems, Digital Advertising	DLRM-Small (Deep Learning Recommendation Model)	Time-to-Result (Time to reach a target AUC)	Datacenter CPUs/GPUs with high memory bandwidth (HBM) due to massive embedding tables, and highly optimized network I/O.	Stresses memory access and sparse feature embedding computations due to the large, sparse Criteo 1TB dataset. Represents a common commercial workload.
AlgoPerf	FastMRI	k-space MRI Reconstruction	Medical Imaging, Healthcare Diagnostics	U-Net (Convolutional Encoder-Decoder)	Time-to-Result (Time to reach a target PSNR / SSIM)	High-Performance GPUs and dedicated AI accelerators, as the model must run with high accuracy and low latency for clinical use.	Focuses on accelerating the image formation process from raw MRI data. U-Net is a standard model for semantic segmentation and image-to-image translation tasks.
AlgoPerf	ImageNet	Image Classification	General-purpose Computer Vision	ResNet-50 and Vision Transformer (ViT) variants	Time-to-Result (Time to reach a target Top-1 Accuracy	General-Purpose GPUs (Training/Inference), Edge Devices, and Mobile SoCs, as it is a widely-used test across all compute scales.	The quintessential computer vision workload. Includes two major architecture types (CNN and Transformer) to test algorithm generalizability.
AlgoPerf	LibriSpeech	Speech Recognition / ASR (Automatic Speech Recognition)	Voice Assistants, Transcription Services	Conformer and DeepSpeech variants	Time-to-Result (Time to reach a target Word Error Rate (WER))	Datacenter/Cloud GPUs (for large-scale ASR), Edge/Mobile Processors (for on-device assistants).	Tests algorithms on sequential data. Conformer is a hybrid CNN/Transformer architecture common in modern ASR.
AlgoPerf	OGBG	Graph Property Prediction	Scientific Machine Learning, Drug Discovery, Social Networks	GNN (Graph Neural Network)	Time-to-Result (Time to reach a target ROC-AUC)	Datacenter CPUs/GPUs with high-speed interconnects due to the irregular, sparse nature of graph-structured data.	Uses the Open Graph Benchmark (OGB) dataset. This workload stresses algorithms in domains that rely on non-Euclidean data structures.
AlgoPerf	WMT	Machine Translation (En-De)	Natural Language Processing (NLP), Global Communication	Transformer (Base Architecture)	Time-to-Result (Time to reach a target BLEU Score)	Datacenter CPUs/GPUs with specialized Tensor Cores for efficient processing of the Transformer's self-attention mechanism.	A standard, large-scale sequence-to-sequence task, famous for being the original domain of the Transformer architecture.
MLCommons AILuminate							
AILuminate Safety v1.0	System Under Test (SUT) (Any LLM-based general-purpose chat system)	Assess Baseline AI Safety and Reliability	Pre-deployment Validation, Regulatory Compliance, Vendor Comparison	LLMs and AI Chat Systems (Text-to-Text), potentially with guardrails/filters	Overall Safety Grade (5-tier scale: Poor to Excellent), Violation Rate (% of unsafe responses), Per-Hazard Performance	The AI System itself (typically hosted in a Datacenter/Cloud) is the system under test (SUT). The evaluation is performed by a separate, specialized Safety Evaluator Model (often a tuned LLM ensemble).	Assesses safety against 12 Hazard Categories (e.g., Violent Crimes, Hate, Suicide & Self-Harm). Uses a tuned ensemble of safety evaluator models for grading. Focuses on single-turn, content-only hazards.
AILuminate Jailbreak Benchmark v0.5	System Under Test (SUT) (Any LLM-based general-purpose chat system)	Quantify Resilience to Adversarial "Jailbreak" Attacks	AI Security, Robustness Testing, Defense Mechanism Comparison	LLMs (Text-to-Text) and Vision-Language Models (VLMs) (Text+Image-to-Text)	Resilience Gap (Drop in safety performance from baseline to under-attack), Jailbreak Success Rate	The AI System (SUT) is tested in a Datacenter/Cloud environment. The benchmark focuses on the input (adversarial prompts) and the system's subsequent failure rate under attack conditions.	v0.5 is an initial release establishing the framework. It specifically measures the degradation of safety when a system is subjected to prompts designed to bypass its safety filters ("jailbreaks").

TABLE II: Large Language Model Benchmark Details

Benchmark Name	Model	Task	Application Domain / Use Case	Model Type / Architecture	Metrics / KPIs	Notes / Description
Commercial/Proprietary LLMs (API/Systems)						
LLM Inference	Claude 3.5 Haiku 20241022	Generative AI	General Purpose, Light Reasoning	Large Transformer (Proprietary)	TTFT, TPOT, Throughput, MMLU (Quality)	A faster, smaller version in the Claude 3.5 family.
LLM Inference	Claude 3.5 Sonnet 20241022	Generative AI	Complex Reasoning, Data Processing	Large Transformer (Proprietary)	TTFT, TPOT, Throughput, MMLU (Quality)	Mid-tier model focusing on balance of speed and intelligence.
LLM Inference	Mistral Large 2402 Moderated	Generative AI	Enterprise Chatbots, Content Moderation	MoE/Dense Transformer (Proprietary)	TTFT, TPOT, Throughput, Safety Index	Flagged as moderated; emphasis on safety and reliable output.
LLM Inference	Amazon Nova Lite v1.0	Generative AI	AWS Services, Embedded Use Cases	Large Transformer (Proprietary)	Latency, Throughput, Cost/Token	Lightweight, cloud-optimized model.
LLM Inference	Gemini 1.5 Pro (API, with option)	Generative AI / Multimodal	Long Context, Multi-Source Reasoning	MoE/Dense Transformer (Proprietary, Multimodal)	TTFT, Throughput, Latency, RAG/Context Recall	Known for its massive context window.
LLM Inference	Gemini 2.0 Flash 001	Generative AI / Multimodal	High-Speed Chat, Real-time Tasks	Dense Transformer (Proprietary, Multimodal)	p99 Latency, Throughput	Focuses on speed and efficiency for low-latency needs.
LLM Inference	Gemini 2.0 Flash Lite	Generative AI	Edge/Client-Side Inference	Dense Transformer (Proprietary, Small)	Energy Efficiency, Latency	Highly optimized for resource-constrained environments.
LLM Inference	GPT-4o	Generative AI / Multimodal	Real-time Conversation, Vision Integration	Dense Transformer (Proprietary, Multimodal)	TTFT, TPOT, Low-Latency Response	All-in-one model for low-latency multimodal interactions.
LLM Inference	GPT-4o mini	Generative AI	Quick, Cost-Effective Tasks	Dense Transformer (Proprietary, Small)	Cost/Token, Throughput	Optimized for efficiency and scaling simple tasks.
LLM Inference	Minustral 8B 24.10 (API)	Generative AI	General Text Generation	MoE/Dense Transformer (Proprietary)	Latency, Throughput	Represents a competitive, smaller model in a commercial API.
Open-Source/Bare Models (Used for Training or Deployment)						
LLM Inference	Minustral 8B 24.10 Moderation	Generative AI	General Text Generation, Safety Research	MoE/Dense Transformer (Open-weights)	Latency, Safety Compliance	Open-weight version with a focus on safety.
LLM Inference	Gemma 2 9b	Generative AI	Fine-tuning, Edge Deployment	Dense Transformer (Open-weights)	Perplexity, MMLU, Throughput	Smaller model from the Gemma family, good for fine-tuning.
LLM Inference	Phi 3.5 MoE Instruct	Generative AI	Instruction Following, Small Scale Reasoning	MoE (Open-weights, Small)	MMLU, HumanEval (Code)	Instruction-tuned, likely using a small Mixture-of-Experts.
LLM Inference	Phi 4	Generative AI	Research, Prototyping	Dense Transformer (Open-weights, Small)	Perplexity, BLEU (Generation)	Successor in the Phi family, typically very small.
LLM Inference	Athene V2 Chat Hf	Generative AI	Open Chatbot Deployment	Dense Transformer (Open-weights, Fine-tuned)	TTFT, TPOT, Chat Metrics	An instruction-tuned model from the Hugging Face ecosystem.
LLM Inference	Aya Expanse 8B Hf	Generative AI	Multilingual Tasks, Text Translation	Dense Transformer (Open-weights)	BLEU (Translation), Accuracy	Focused on broad language coverage.
LLM Inference	Cohere C4Ai Command A 03 2025 Hf	Generative AI	Enterprise RAG, Instruction Following	Dense Transformer (Open-weights)	Contextual Recall, RAG Latency	Cohere model variant used in the Hugging Face ecosystem.
LLM Inference	Llama 3.1 405B Instruct	Generative AI	State-of-the-Art Reasoning, Long Context	Dense Transformer (Open-weights)	TTFT, Throughput, MMLU	An extremely large, cutting-edge open-weight model (used in MLPerf).
LLM Inference	Llama 3.1 8b Instruct FP8	Generative AI	Edge/Quantized Deployment	Dense Transformer (Quantized)	Inference Accuracy, Memory Footprint	Highly optimized for efficient computation using 8-bit precision.
LLM Inference	Llama 3 1 Tulu 3 8B Hf	Generative AI	General Chat, Fine-tuning Research	Dense Transformer (Open-weights, Fine-tuned)	Alpaca Eval, Human Preference	A variant of Llama tuned for instruction following.
LLM Inference	Mistralai Mistral Large 2402	Generative AI	Complex Reasoning, RAG	MoE/Dense Transformer (Open-weights)	TTFT, TPOT, MMLU	Open-weight version of Mistral's flagship model.
LLM Inference	Olmo 2 0325 32b Instruct	Generative AI	Research, Reproducible AI	Dense Transformer (Open-weights)	Perplexity, Training Speed	High-parameter model focused on openness and research.
LLM Inference	Olmo 2 1124 13B Instruct Hf	Generative AI	Instruction Following, General Chat	Dense Transformer (Open-weights)	TTFT, Throughput	Smaller, instruction-tuned version of the Olmo family.
LLM Inference	Phi 3.5 Mini Instruct	Generative AI	Mobile/Edge Inference, Simple Tasks	Dense Transformer (Open-weights, Small)	Latency, MMLU	Ultra-small model optimized for fast responses.
LLM Inference	Qwen1 5 110B Chat Hf	Generative AI	Multi-Language Chat, High Accuracy	Dense Transformer (Open-weights)	C-Eval, MMLU, Throughput	High-parameter model known for strong Chinese/general performance.
LLM Inference	Yi 1 5 34B Chat Hf	Generative AI	General Purpose, Instruction Following	Dense Transformer (Open-weights)	MMLU, C-Eval, Latency	Mid-to-large size model focusing on quality chat performance.
LLM Inference	Ai21Labs Ai21 Jamba Large 1.5 Azure	Generative AI	Cloud Deployment, Enterprise Apps	Hybrid MoE/Dense Transformer	Throughput, Latency	A large model known for its hybrid architecture, deployed via Azure.
LLM Inference	Google Gemma 3 27B It Hf Nebius	Generative AI	Cloud Deployment, Fine-tuning	Dense Transformer (Open-weights, Fine-tuned)	TTFT, TPOT, Cloud Efficiency	Gemma model deployed on the Nebius cloud platform.
LLM Inference	Llama 3.3 70B Instruct Turbo Together	Generative AI	Fast, High-Quality Instruction Following	Dense Transformer (Open-weights)	Latency, Throughput, Cost	A large model optimized for speed via the Together API.
LLM Inference	Mistral Large 24.11	Generative AI	Enterprise AI, High Performance	MoE/Dense Transformer (Open-weights)	Throughput, MMLU, Reasoning	A very recent high-performance model.
LLM Inference	Qwq 32B Hf	Generative AI	General Purpose, Instruction Following	Dense Transformer (Open-weights)	Latency, Throughput	A mid-sized model in the open-weight ecosystem.
LLM Inference	OLMo 7b 0724 Instruct	Generative AI	Research, Instruction Following	Dense Transformer (Open-weights)	Perplexity, Speed	Smaller, instruction-tuned model for general tasks.

TABLE III: Ontology Table for Selected AI Science Benchmarks.

(For detailed view of the Radar Charts, see [24].)

Ratings	Name	Domain	Models	Metrics	Citation
	ClimateLearn - Weather Forcasting	Climate & Earth Science	CNN baselines, ResNet variants	RMSE, Anomaly correlation	[28]
	ClimateLearn - Downscaling	Climate & Earth Science	CNN baselines, ResNet variants	RMSE, Anomaly correlation	[28]
	ClimateLearn - Climate Projection	Climate & Earth Science	CNN baselines, ResNet variants	RMSE, Anomaly correlation	[28]
	MLCommons Science - CloudMask	Climate & Earth Science	CNN, GNN, Transformer	MAE, Accuracy, Speedup vs simulation	[29]
	MLCommons Science - Earthquake	Climate & Earth Science	CNN, GNN, Transformer	MAE, Accuracy, Speedup vs simulation	[29]
	MLCommons Science - Candle UNO	Biology & Medicine	CNN, GNN, Transformer	MAE, Accuracy, Speedup vs simulation	[29]
	MLCommons Science - STEMDL	Materials Science	CNN, GNN, Transformer	MAE, Accuracy, Speedup vs simulation	[29]
	ARC-Challenge (Advanced Reasoning Challenge)	Computational Science & AI	GPT-4, Claude	Accuracy	[30]
	MOLGEN	Chemistry	MolGen	Validity%, Novelty%, QED, Docking score, penalized logP	[31]
	Open Graph Benchmark (OGB) - Biology	Biology & Medicine	GCN, GraphSAGE, GAT	Accuracy, ROC-AUC	[32]
	LLMs for Crop Science	Climate & Earth Science	GPT-3.5, GPT-4, Claude-3-opus, Qwen-max, LLama3-8B, InternLM2-7B, Qwen1.5-7B	Accuracy, F1 score	[33]
	SciCode	Computational Science & AI	Claude3.5-Sonnet	Solve rate (%)	[34]
	CaloChallenge 2022	High Energy Physics	VAE variants, GAN variants, Normalizing flows, Diffusion models	Histogram similarity, Classifier AUC, Generation latency	[35]
	PDEBench	Computational Science & AI, Climate & Earth Science, Mathematics	FNO, U-Net, PINN, Gradient-Based inverse methods	RMSE, boundary RMSE, Fourier RMSE	[36]
	Urban Data Layer (UDL) - PM2.5 Concentration Prediction	Climate & Earth Science	Baseline regression/classification pipelines	Task-specific accuracy or RMSE	[37]
	Urban Data Layer (UDL) - Built-up Area Classification	Climate & Earth Science	Baseline regression/classification pipelines	Task-specific accuracy or RMSE	[37]
	Urban Data Layer (UDL) - Administrative Boundaries Identification	Climate & Earth Science	Baseline regression/classification pipelines	Task-specific accuracy or RMSE	[37]

Continued on next page

TABLE III: Ontology Table for Selected AI Science Benchmarks (cont.).

Ratings	Name	Domain	Models	Metrics	Citation
	Urban Data Layer (UDL) - El Nino Anomaly Detection	Climate & Earth Science	Baseline regression/classification pipelines	Task-specific accuracy or RMSE	[37]
	SPIQA (LLM)	Computational Science & AI	LLaVA, MiniGPT-4, Owl-LLM adapter variants	Accuracy, F1 score	[38]
	MLCommons Medical AI - Pancreas Segmentation (DFCI)	Biology & Medicine	MedPerf-validated CNNs, GaNDLF workflows	ROC AUC, Accuracy, Fairness metrics	[39]
	MLCommons Medical AI - Brain Tumor Segmentation (BraTS)	Biology & Medicine	MedPerf-validated CNNs, GaNDLF workflows	ROC AUC, Accuracy, Fairness metrics	[39]
	MLCommons Medical AI - Surgical Workflow Phase Recognition (SurgMLCube)	Biology & Medicine	MedPerf-validated CNNs, GaNDLF workflows	ROC AUC, Accuracy, Fairness metrics	[39]
	SeafloorAI	Climate & Earth Science	SegFormer, ViLT-style multimodal models	Segmentation pixel accuracy, QA accuracy	[40]
	SeafloorGenAI	Climate & Earth Science	SegFormer, ViLT-style multimodal models	Segmentation pixel accuracy, QA accuracy	[40]
	GeSS - Track Pileup	High Energy Physics	GCN, EGNN, DimeNet++	Accuracy, RMSE, OOD robustness delta	[41]
	GeSS - Track Signal	High Energy Physics	GCN, EGNN, DimeNet++	Accuracy, RMSE, OOD robustness delta	[41]
	GeSS - DrugOOD	Biology & Medicine	GCN, EGNN, DimeNet++	Accuracy, RMSE, OOD robustness delta	[41]
	GeSS - QMOF	Materials Science	GCN, EGNN, DimeNet++	Accuracy, RMSE, OOD robustness delta	[41]
	OCP (Open Catalyst Project)	Chemistry, Materials Science	CGCNN, SchNet, DimeNet++, GemNet-OC	MAE (energy), MAE (force)	[42, 43]
	Jet Classification	High Energy Physics	Keras DNN, QKeras quantized DNN	Accuracy, AUC	[44]
	Irregular Sensor Data Compression	High Energy Physics	Autoencoder, Quantized autoencoder	MSE, Compression ratio	[44]
	MLPerf HPC - Cosmoflow	High Energy Physics	CosmoFlow, DeepCAM, OpenCatalyst	Training time, Accuracy, GPU utilization	[45]
	MLPerf HPC - DeepCAM	Climate & Earth Science	DeepCAM	Training time, Accuracy, GPU utilization	[45]
	MLPerf HPC - Open Catalyst Project DimeNet++	Chemistry	DeepCAM	Training time, Accuracy, GPU utilization	[45]
	MLPerf HPC - OpenFold	Biology & Medicine	DeepCAM	Training time, Accuracy, GPU utilization	[45]

Continued on next page

TABLE III: Ontology Table for Selected AI Science Benchmarks (cont.).

Ratings	Name	Domain	Models	Metrics	Citation
	HDR ML Anomaly Challenge - Gravitational Waves	High Energy Physics	Deep latent CNNs, Autoencoders	ROC-AUC, Precision/Recall	[46]
	SuperCon3D - Property Prediction	Materials Science	SODNet, DiffCSP-SC	MAE (Tc), Validity of generated structures	[47]
	SuperCon3D - Inverse Crystal Structure Generation	Materials Science	SODNet, DiffCSP-SC	MAE (Tc), Validity of generated structures	[47]
	BaisBench (Biological AI Scientist Benchmark) - Question Answering	Biology & Medicine	LLM-based AI scientist agents	Annotation accuracy, QA accuracy	[48]
	BaisBench (Biological AI Scientist Benchmark) - Cell Type Annotation	Biology & Medicine	LLM-based AI scientist agents	Annotation accuracy, QA accuracy	[48]
	The Well	Biology & Medicine, Computational Science & AI, High Energy Physics	FNO baselines, U-Net baselines	Dataset size, Domain breadth	[49]
	MMLU (Massive Multitask Language Understanding)	Computational Science & AI	GPT-4o, Gemini 1.5 Pro, o1, DeepSeek-R1	Accuracy	[50]
	SatImgNet	Climate & Earth Science	CLIP, BLIP, ALBEF	Accuracy	[51]
	GPQA Diamond	Biology & Medicine, Chemistry, High Energy Physics	o1, DeepSeek-R1	Accuracy	[52]
	PRM800K	Mathematics	GPT-4	Accuracy	[53]
	FEABench (Finite Element Analysis Benchmark): Evaluating Language Models on Multiphysics Reasoning Ability	Mathematics	FEniCS, deal.II	Solve time, Error norm	[54]
	Neural Architecture Codesign for Fast Physics Applications	High Energy Physics	NAC-based BraggNN, NAC-optimized Deep Sets (jet)	Accuracy, Latency, Resource utilization	[55]
	Delta Squared-DFT	Chemistry, Materials Science	Delta Squared-ML correction networks, Kernel ridge regression	Mean Absolute Error (eV), Energy ranking accuracy	[56]
	HDR ML Anomaly Challenge - Sea Level Rise	Climate & Earth Science	CNNs, RNNs, Transformers	ROC-AUC, Precision/Recall	[46]
	Vocal Call Locator (VCL)	Biology & Medicine	CNN-based SSL models	Localization error (cm), Recall/Precision	[57]
	MassSpecGym - De novo molecule generation	Chemistry	Graph-based generative models, Retrieval baselines	Structure accuracy, Retrieval precision, Simulation MSE	[58]
	MassSpecGym - Molecule Retrieval	Chemistry	Graph-based generative models, Retrieval baselines	Structure accuracy, Retrieval precision, Simulation MSE	[58]

Continued on next page

TABLE III: Ontology Table for Selected AI Science Benchmarks (cont.).

Ratings	Name	Domain	Models	Metrics	Citation
	MassSpecGym - Spectrum Simulation	Chemistry	Graph-based generative models, Retrieval baselines	Structure accuracy, Retrieval precision, Simulation MSE	[58]
	SPIQA (Scientific Paper Image Question Answering)	Computational Science & AI	Chain-of-Thought models, Multimodal QA systems	Accuracy, F1 score	[59]
	GPQA: A Graduate-Level Google-Proof Question and Answer Benchmark	Biology & Medicine, High Energy Physics, Chemistry	GPT-4 baseline	Accuracy	[60]
	MedQA	Biology & Medicine	Neural reader, Retrieval-based QA systems	Accuracy	[61]
	Single Qubit Readout on QICK System	Computational Science & AI	hls4ml quantized NN	Accuracy, Latency	[62]
	CFDBench (Fluid Dynamics)	Mathematics	FNO, DeepONet, U-Net	L2 error, MAE	[63]
	CURIE (Scientific Long-Context Understanding, Reasoning and Information Extraction)	Materials Science, High Energy Physics, Biology & Medicine, Chemistry, Climate & Earth Science	unkown	Accuracy	[64]
	Smart Pixels for LHC	High Energy Physics	2-layer pixel NN	Data rejection rate, Power per pixel	[65]
	LHC New Physics Dataset	High Energy Physics	Autoencoder, Variational autoencoder, Isolation forest	ROC-AUC, Detection efficiency	[66]
	Quantum Computing Benchmarks (QML)	Computational Science & AI	IBM Q, IonQ, AQT@LBNL	Fidelity, Success probability	[67]
	Ultrafast jet classification at the HL-LHC	High Energy Physics	MLP, Deep Sets, Interaction Network	Accuracy, Latency, Resource utilization	[68]
	HEDM (BraggNN)	Materials Science	BraggNN	Localization accuracy, Inference time	[69]
	4D-STEM	Materials Science	CNN models (prototype)	Classification accuracy, Throughput	[70]
	Beam Control	High Energy Physics	DDPG, PPO (planned)	Stability, Control loss	[44, 71]
	Intelligent experiments through real-time AI	High Energy Physics	Bipartite Graph Network with Set Transformers (BGN-ST), GarNet (edge-classifier)	Accuracy (charm and beauty detection), Latency (micros), Resource utilization (LUT/FF/BRAM/DSP)	[72]
	HDR ML Anomaly Challenge - Butterfly	Biology & Medicine	CNN-based detectors	Classification accuracy, F1 score	[46]
	DUNE	High Energy Physics	CNN, LSTM (planned)	Detection efficiency, Latency	[73]

Continued on next page

TABLE III: Ontology Table for Selected AI Science Benchmarks (cont.).

Ratings	Name	Domain	Models	Metrics	Citation
	FrontierMath	Mathematics	unknown	Accuracy	[74]
	AIME (American Invitational Mathematics Examination)	Mathematics	unknown	Accuracy	[75]
	Quench detection	High Energy Physics	Autoencoder, RL agents (in development)	ROC-AUC, Detection latency	[76]
	Materials Project	Materials Science	Automatminer, Crystal Graph Neural Networks	MAE, R^2	[77]
	In-Situ High-Speed Computer Vision	High Energy Physics	CNN	Accuracy, FPS	[78]

C. Technical aspects of AI Benchmarks

In addition to discoverability challenges, there are also technical issues that need to be addressed in dealing with democratization and AI benchmark carpentry.

1) Workflows: There are many workflow frameworks that can support the AI Benchmark Workflow. Two of them are the Compute Coordinator and the Experiment Executer; they can be used in conjunction or separately [79]. The Compute Coordinator allows hybrid infrastructure access from the benchmark application, while the Experiment Executor allows the repeated execution of templated benchmarks. Both produce results in a structured fashion so they can be combined from multiple experiments and multiple infrastructures in order to support the FAIR principles.

2) Containerization: Benchmarking on HPC and even smaller machines can be simplified by providing containerized environments which not only enable easy deployment, but also can harmonize execution by providing stable operating system and software environments. In addition to portable makefiles, the uniform generation of containers can be leveraged between applications. Although docker is today widely used to containerize applications, on HPC systems we find that limited root access on many HPC systems led to the development of apptainers. Hence, AI benchmarking carpentry should include the development of software in apptainers directly or converting Docker containers to apptainers.

3) System-Dependent Software and Deployment Variability: Benchmarking can be complex if the software, libraries and infrastructure differ across systems. To support coordinated benchmarking across different machines, we have introduced a templates hybrid reusable computational analytics workflow management framework with cloudmesh. This framework has been be applied to multiple Deep Learning MLCommons Applications. The details are explained in [79]. Utilizing such workflow systems promotes adaptation as deployment and execution is typically included in the workflow specifications. However, it can also address adaptation and modifications to future improvements and porting to different hardware as a working template is already provided..

4) Logging and Monitoring: A variety of logging frameworks exist for AI Benchmark logging. This includes logging tools such as MLPerf logging. While such tools provide elementary logging features, their outputs are not human readable and require post processing. This is also an issue when running applications in interactive mode during debugging phases. For this reason, we have provided Cloudmesh-stopwatch that not only allows human readable format, but also allows automatic MLPerf logging (if desired) with a single line change in the code. Cloudmesh stopwatch supports Python, shell, and batch script execution, and employs a consistent log format across all three.

In general, we distinguish between four types of monitoring: (a) Infrastructure Monitoring, (b) Application Monitoring, (c) Training Monitoring, and (d) Model-Level Monitoring. A wide range of tools exists for each type, making it essential to identify those that provide effective functionality while remaining easy to use. TensorBoard is one example.

5) Profiling and Performance Analysis: Profiling is the process of measuring a program's performance in association with the locations in the source code in order to reveal where resources (e.g., time and memory) are spent during execution. Profiling is important in AI benchmarking for the following reasons:

- Profiling helps explain why a particular method or implementation variant is faster than another.
- Profiling helps support fair and reproducible benchmarking.
- Profiling can distinguish between the essential computations and extraneous overheads.
- In a heterogeneous system, profiling can identify which components (e.g., CPU or GPU's CUDA cores vs. tensor cores) are being used by different parts of the application.
- Profiling can identify which specific library kernels are being used by different parts of the application.

Table IV provides a list of profiling tools that are useful for analysis of deep learning applications.

It is important to note that the tooling and services exist for supporting different levels of infrastructures. This includes examples for framework-level, system-level (including CPU and GPU), kernel-level, compiler-level, communication-level, and cloud-level.

Furthermore, we aim here to provide comprehensive coverage of the AI profiling stack, which affords users the insights into cross-vendor and cross-platform capabilities and offerings, and also provide key analysis of features of the said tools and services.

We believe it is essential to increase awareness and use of profiling tools through AI benchmarking efforts, enabling a better understanding of bottlenecks in AI applications. Additionally, we need to educate the community about policy limitations that may implicitly restrict specific profiling tools. As discussed previously, one such policy restriction is that not all profiling information is available for energy benchmarks. Such restrictions may also be in place for additional hardware profiling measures.

Lastly, we need to educate the community about the *performance impact* of profiling costs to avoid over-profiling. Therefore, it makes sense that AI benchmarks should be able to choose the level of profiling selectively. This information is vital to support the FAIR principles and ensure that benchmarks are comparable.

D. GPU Benchmarking and its Variability

Modern scientific applications frequently require peta- or exascale levels of compute to model topics with high fidelity. To meet these demands in reasonable timeframes, scientists

TABLE IV: Summary of Example Profiling Tools Useful for Deep Learning and AI Workloads

Tool / Category	Vendor / Maintainer	Level / Primary Use Case	Key Features and Capabilities
Framework Profilers			
PyTorch Profiler [80]	Meta	Framework-level	Records CPU/GPU activities, memory usage, and operator timings; integrates with TensorBoard and Perfetto; useful for training optimization and layer timing.
TensorBoard / TensorFlow Profiler [80]	Google	Framework-level	Visualizes input pipelines, GPU kernels, and op-level timings; includes memory and device utilization tracing; supports bottleneck analysis.
torch.utils.bottleneck [81]	Meta	Framework-level	Combines autograd and Python profilers for quick bottleneck diagnostics.
JAX Profiler [82]	Google	Framework-level	Works with TensorBoard to trace XLA compilation, HLO graphs, and TPU/GPU runtime performance.
NVIDIA DLProf [83]	NVIDIA	Framework-level (GPU-focused)	High-level view of deep learning layers and operations; integrates with TensorBoard DLProf plugin.
Hardware / System Profilers			
Nsight Systems [84]	NVIDIA	System-level	Timeline visualization of CPU–GPU interactions, kernel launch overheads, multi-process analysis, and NCCL tracing.
Nsight Compute [85]	NVIDIA	Kernel-level	Detailed GPU kernel performance metrics: memory throughput, Tensor Core utilization, occupancy, and roofline analysis.
nvprof (deprecated) [86]	NVIDIA	GPU-level	Legacy command-line CUDA profiler, replaced by Nsight tools.
VTune Profiler [87]	Intel	CPU/System-level	Hotspot analysis, vectorization, threading efficiency, and CPU performance bottlenecks.
omnitrace / rocprof / rocm-smi [88]	AMD	GPU-level	Profiling and monitoring for AMD GPUs: kernel execution metrics, power, and temperature.
HPCToolkit [89]	Rice University	System-level (CPU+GPU)	Hierarchical performance profiling, time attribution to calling context, supports CUDA and HIP.
TAU [90]	University of Oregon	System-level (CPU+GPU+MPI)	Multi-level performance analysis, MPI integration, supports heterogeneous systems.
Perfetto [91]	Google (Open Source)	System-level	High-resolution trace visualization, interoperable with PyTorch/TensorFlow profiler exports.
PAPI [92]	University of Tennessee	Hardware counter interface	Provides access to CPU/GPU performance counters for integration with other profiling tools or custom instrumentation.
Compiler / Graph Profilers			
XLA Profiler [93]	Google	Compiler-level (XLA)	Profiles XLA-compiled operations and execution times; supports JAX/TF and TPU/GPU workloads.
TorchDynamo / TorchInductor Debug Tools [94]	Meta	Compiler-level (PyTorch 2.x)	Analyzes graph fusion, compiler optimizations, and operator performance of compiled PyTorch models.
Triton Profiler [95]	OpenAI	Kernel-level (Custom Kernels)	Reports kernel execution time, register usage, and occupancy for custom Triton GPU kernels.
Communication / Distributed Profilers			
NCCL Profiler [96]	NVIDIA	Communication-level	Profiles NCCL collective communication operations (e.g., all-reduce, broadcast); timeline visualization of multi-GPU communication.
AWS SageMaker Debugger / Azure Profiler [97, 98]	AWS / Microsoft	Cloud-level	Distributed GPU/CPU monitoring, training metric collection, and profiling at cloud scale.
Weights & Biases, Comet, MLflow [99, 100, 101]	Multiple Vendors	Experiment / Cloud-level	Logs performance traces, GPU utilization, integrates with PyTorch and TensorFlow profilers for real-time monitoring.
System & Memory Profilers			
Torch / TensorFlow Memory Tools [102]	Meta / Google	Framework-level (Memory)	Reports GPU memory allocation, fragmentation, and utilization trends for debugging memory bottlenecks.
Python Profilers (cProfile, py-spy) [103]	Python Community	CPU-level	Measures Python-level overhead and I/O performance; used for diagnosing data preprocessing bottlenecks.

and researchers typically run these workloads on massively parallel systems such as GPUs. For example, workloads such as graph analytics [104, 105], scientific computing [106, 107, 108, 109], ML [110, 111, 112, 113, 114, 115, 116, 117] heavily utilize GPUs. Increasingly, ML is also impacting scientific applications [118, 119, 120, 121, 122] by replacing or supplementing traditional computing methods in application domains like molecular dynamics (e.g., DeePMD [123, 124]), protein folding (e.g., OpenFold2 [125]), and scientific AI models (e.g., AuroraGPT [2]). However, given the scale of data these workloads operate on and the large size of the workloads themselves, they typically must partition their work across many GPUs.

Given their widespread use and trend towards many GPU applications, it is desirable from a benchmark carpentry perspective to make GPU experiments repeatable and consistent. For traditional HPC systems composed of multiple CPUs, prior work showed that this was difficult to achieve: application performance varied by up to 20%, even for CPUs with the same architecture and vendor SKU (Stock-Keeping Unit) [126, 127, 128, 129, 130, 131]. This variation occurs due to the manufacturing process and the chip's power constraints [128, 132]. Such dynamic behavior makes it challenging for repeatable experiments, and can lead to resource underutilization. Unfortunately, similar issues also arise in modern systems composed of many GPUs. Recent work has demonstrated that GPU-rich systems suffer from significant performance variability [133, 134, 135, 132, 136, 137, 138, 139, 140].

For example, Sinha, et al. examined variability across five modern GPU-rich clusters with a variety of sizes, cooling approaches, and GPU vendors [137]. They found that applications exhibited performance variability of 8% on average (max 22%) with outliers up to 1.5× slower than the median GPU. Moreover, these results were consistent over time (i.e., not transient) and were unaffected by GPU vendors or cooling type. Interestingly, this performance variability was also application-specific: the more compute-intensive the application was, the more performance variability the application observed due to effects of the GPU's power management algorithm (e.g., Dynamic Voltage & Frequency Scaling—DVFS). Furthermore, performance variability is getting worse as transistors continue scaling [141].

Although the impact of performance variability is significant for single-GPU workloads, it is even larger for multi-GPU workloads. Currently, GPU-rich systems focus on scheduling work to minimize the number of nodes an application requests, without considering variability. In the five clusters from this prior work, users asking for 4 GPUs for a given application would get a slower GPU allocated to them between 22% (Sandia's Vortex cluster) and 50% (TACC's Longhorn cluster [142, 143]) of the time. Thus, users are likely to get a slow GPU frequently, especially since modern scientific workloads often request 64 or more GPUs for a given experiment. This can lead to significant resource under-utilization for multi-GPU jobs since all of them must wait for the slowest one to complete due to the bulk synchronous programming (BSP) model used in many data-parallel workloads [144]. Accordingly, it is imperative for users to be aware of the impact of performance variability on their experiments, and for benchmark carpentry to propose solutions to minimize its effects.

Although GPU-rich systems are likely to suffer from performance variability for the foreseeable future, there are several steps various stakeholders, such as users, maintainers, and system designers, can take to reduce the impact on obtaining statistically significant results in existing systems. First, cluster operators can perform periodic performance-variability benchmarking to identify underperforming GPUs and perform targeted maintenance on them. Likewise, users can perform similar benchmarking to identify GPUs that behave similarly, and then use blacklisting or other scheduling approaches to attempt to schedule work on GPUs with similar performance variability profiles. However, doing so can be time- and labor-intensive for clusters with thousands or more GPUs (though it is a one-time cost, since a GPU's performance variability is consistent over time). Thus, a more scalable, dynamic approach is to redesign job-scheduling policies for GPU clusters to account for performance variability when making scheduling decisions. Recent work has shown that embracing performance variability can transparently and significantly improve job completion time, makespan, and GPU utilization [145]. Finally, since performance variability is application-specific, we recommend that new, unprofiled applications either rely on other applications with similar profiles as proxies [146] or be profiled during their first execution on a new cluster to determine their sensitivity to performance variability.

In terms of democratizing the availability of multi-GPU systems there are several barriers to overcome, these are the cost, access, skills and complexity The cost barrier means that the large-scale systems are affordable only to national labs and major corporations. Consequently, the access is usually restricted to the staff of these organizations. Using multi-GPU systems effectively requires specialized knowledge. Users must be trained in containerization technologies, distributed libraries, and orchestration tools that allow applications to scale across many GPUs. There is also a barrier on the conceptual level. The performance of a multi-GPU system is the result of interactions between hardware, interconnects, and software stacks. At present, we lack high-level performance prediction model that can reliably describe how applications behave when running on GPUs. This makes it difficult to plan experiments, determine the required resources and generalize findings.

E. Energy Benchmarking

Energy consumption is a critical component of ML benchmarking. Training and inference with modern AI systems can require enormous computational resources.

TABLE V: Estimated Energy Consumption of GPT Models for Training and Inference (Based on [155, 152, 156, 157, 158, 159]).

Model	Training Energy (MWh)	Inference Energy (per 1M queries, MWh)
GPT-3	∼1,287 [154, 152]	∼50–100
GPT-4	51,773–62,319 [156, 157]	∼600–1,000
GPT-5	>60,000 (estimated) [158, 159]	∼800–1,200
GPT-6	80,000–100,000 (projected) [158]	∼1,000–1,500

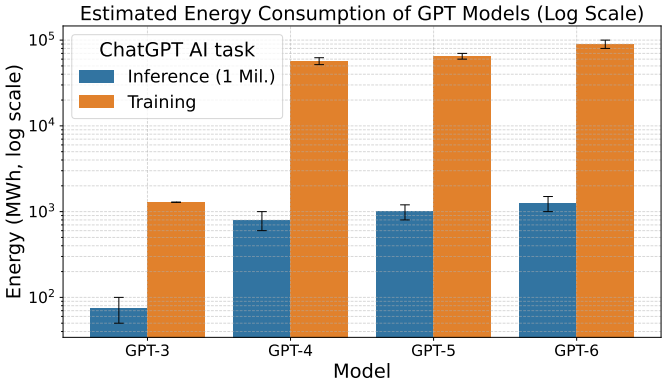

Fig. 1: Energy Consumption for ChatGPT Training and Inferencing 1 Million Queries
(Data for GPT-5 and 6 are estimates).

To illustrate the issue, we have provided in Table V and Figure 1 the energy required to train various ChatGPT models (some of which are estimated as no public data has been released [147, 148], such as GPT-5 and GPT-6). The training of a single large-scale language model (GPT-3) consumes approximately 1,287 MWh placing it in the same range as the annual energy usage of about 130 U.S. households, according to U.S. Energy Information Administration (EIA) statistics on average residential electricity consumption [149, 150, 151, 152, 153, 154].
For the U.S. Department of Energy (DOE) leadership-class machines, such as those hosted at Oak Ridge National Laboratory (see Table VI), we find documented and significant progress toward exascale, but at the cost of increased energy consumption that more than doubled during the last generational upgrade. However, the Peak Performance per energy unit has increased significantly, and compared to Jaguar's initial values, Frontier has improved by a factor of 209, thus becoming relatively more efficient despite overall energy consumption needs.

TABLE VI: Evolution of the Leadership Class Supercomputer at Oak Ridge National Laboratory

Machine	Year	Architecture	R_{max} Scaling	R_{max} PFlops/s	R_{peak} PFlops/s	Power (MW)	R_{max}/Power (PF/MW)
Jaguar[160]	2009	Multi-core CPU	1	1941	2628	7	277.29
Titan[161]	2012	Hybrid CPU/GPU	9.06	17590	27113	9	1954.44
Summit[162]	2017	Hybrid CPU/GPU	76.6	148600	200795	13	11430.77
Frontier[163]	2022	Hybrid CPU/GPU	697.1	1353000	2055717	29	46655.17

*PF=*Theoretical peta–floating-point operations per second*; 1 PF = 10^{15}FLOPS.
R_{max} = maximal LINPACK performance achieved. R_{peak} = theoretical peak performance.

Carbon-emission measurements also help provide a more detailed understanding of associated energy impacts.
If we only focus on traditional benchmarks using metrics such as FLOPS or latency, we provide performance insights but overlook *energy-to-solution*, which measures the total energy required to complete a task. Without perspective, researchers and practitioners focus on optimizing for speed at the expense of sustainability and cost efficiency.
Thus, we believe it is important to make energy benchmarks an important aspect of AI benchmarks. Energy benchmarking ought to address the following:

- Quantify the environmental footprint of AI workloads (carbon emissions, renewable vs. non-renewable energy use).
- Highlight economic tradeoffs in large-scale computing (cloud costs, datacenter efficiency).
- Guide hardware and algorithmic choices towards a more effective architecture.
- Support policy and funding decisions by providing transparent data on sustainability.

Energy-aware benchmarks help ensure that AI development aligns with broader goals of responsible computing, making results reproducible, performant, and economically and environmentally sustainable.
Thus, we see several opportunities. First, we need to make energy benchmarks more prominent and provide materials and tutorials as part of AI benchmark carpentry to educate the community. Second, we must ensure that not only the most expensive hardware, such as leadership-class and hyper-scale data centers, is used, but also medium- and even small-scale hardware, so that democratizing energy benchmarks within the community is easy to implement. This way, measurements of even smaller AI-based scientific applications can integrate energy consumption into their benchmarks, and meaningful comparisons with traditional algorithms that do not use AI can be drawn. Third, we must ensure that energy metrics and logs can be accessed and uniformly integrated into the AI benchmarks.
1) AI Energy Benchmark Carpentry: To support AI energy benchmark carpentry efforts, we need to address the following issues:

- Conduct a relevant survey of existing efforts
- Identify metrics useful for AI benchmarks
- Identify how to leverage existing and create new leaderboards focusing on energy metrics
- Identify simple-to-use blueprints as part of the carpentry efforts that can not only be replicated and reused, but also serve as a basis for newly developed benchmarks.
- Conduct community outreach to offer carpentry tutorials that focus on AI benchmarks instead of just AI software and services.
- Identify how to obtain and integrate meaningful and practical metrics (e.g., data centers may not provide uniform access to energy data) so that energy data

collection and access become part of carpentry efforts.

Strategies to integrate energy into AI benchmarks for carpentry efforts include improving access to metrics, including the creation of logs during runtime that:

- Log ambient temperature and humidity.
- Log sample power at regular intervals or averages over the run.
- Store the logging data in an easy-to-parse format (CSV, JSON, YAML)
- Upload results as artifacts in support of the FAIR principle and make available for comparison.

Next, we discuss some of the aspects that need to be addressed in more detail.

2) Energy Metrics: There are various energy metrics to consider, including metrics that may not historically received attention. It is also important to identify metrics for leaderboards, but they must be obtained in a way that allows fair, informed comparisons. Hence, it is important to document how the experiment should be conducted rather than just referring to the metric. In principle, blueprints should be used and adapted to make comparisons across hardware and software easier. Energy metrics are used across different layers of the AI benchmark infrastructure, which is similar to classical HPC infrastructure. We provide an example of using different metrics on the various layers in Figure 2. Such diagrams should be integrated into the blueprints provided to users to simplify understanding the benchmarks' energy scope.

As part of the energy augmentation, a clear purpose for the benchmark metric should be stated. Such examples should be collected as part of the experiment's metadata so they can be leveraged and serve as a motivator for other benchmarks. In our example from Figure 2, the purpose for each metric is as follows:

1) **Device/Micro-architectural Layer (D_L)**
   - *Energy per flop* or *Energy per inference*: Measures the energy consumed to perform a single computational operation (a floating-point operation or an inference).
   - *Temperature sensors*: *Related Logging (Non-KPI): Inlet and Outlet Temperature Sensors*: Logged because *thermal headroom* directly bounds the safe *Dynamic Voltage and Frequency Scaling (DVFS)* ranges.
2) **Job/System Layer (J_L)**
   - *Kilowatt-hour (kWh)*: The total energy consumed by a specific job or set of jobs over its duration.
   - *Energy–Delay Product (EDP)*: A combined metric of energy and time (energy × delay) used to assess the overall efficiency of a computation. Lower EDP generally indicates better performance and efficiency.
3) **Facilities/Data Center Layer (F_L)**
   - *Power Usage Effectiveness (PUE)*: A ratio that measures how efficiently a data center uses energy. An ideal PUE is 1.0 (meaning the IT equipment uses all energy). *Data Center Infrastructure Efficiency (DCiE)*: The reciprocal of PUE, expressed as a percentage. It shows the percentage of total data center energy used by IT equipment.

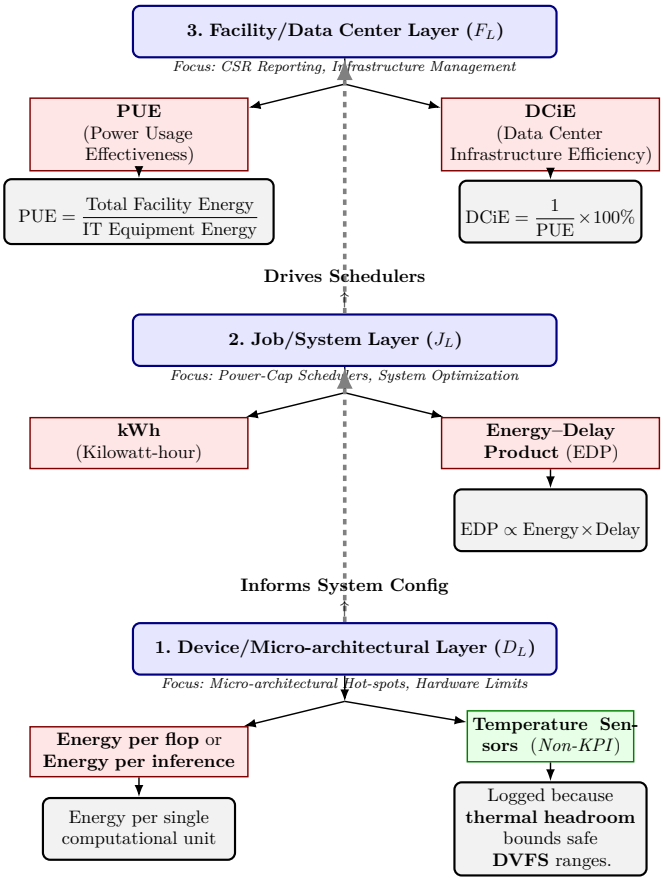

Fig. 2: Illustration of an Example for Metrics as Used in the Layered System Architecture for Large-Scale AI Benchmarking.

This tiered structure, along with a detailed purpose statement for each metric, allows for meaningful comparisons and decision-making at every level of the computing infrastructure.

To identify commonly used metrics, we conducted an initial survey of tools and benchmarks related to energy, which we present in Table VII, while listing their typical benchmark use.

Common requirements for such metrics include obtaining measurements at low cost, sharing results with metadata augmentations, and integrating them into potential leaderboards. We believe we have to go beyond established leaderboards such as *Green500* and the *MLPerf Power*, which already influence processor road-maps and procurement calls [164, 165], to raise awareness of the energy impact on real-world scientific applications.

TABLE VII: Energy- or Carbon-Efficiency (B)enchmarks and (T)ools used in Scientific-HPC research.

(B)enchmark or (T)ool		Core metric(s)	Typical Benchmarking Use
Benchmark			
B SPECpower_ssj2008	[19]	W/transaction; ops/W	Enterprise-server rankings; ENERGY STAR compliance
B SPEC SERT[2]	[166]	Server-Efficiency-Rating = kWh + perf	EU Lot 9 certification; vendor datasheets
B TPC-Energy	[167]	Wh/DB phase	OLTP/warehouse energy cost studies
B JouleSort	[168]	records/J	Storage-I/O contests; I/O-stack tuning
B Green500	[20]	GFLOPS/W (HPL or HPL-AI)	Global supercomputer energy ranking
B HPCG-Power	[169]	GFLOPS/W (HPCG)	Memory-bound tuning; procurement add-on to TOP500
B HPL-MxP (HPL-AI)	[170]	mixed-precision GFLOPS/W	GPU/TPU evaluation for AI-optimised LINPACK
B MLPerf Power	[171]	J; avg W; J/sample; J/epoch	Official energy track for MLPerf submissions
B MLPerf Tiny	[172]	μJ/inference (MCU)	Edge-AI board comparison; ultra-low-power design
B CoreMark-PRO Power	[173]	iterations/s/W (SoC)	Pre-silicon DVFS sweeps; embedded RFPs
B UL Procyon AI Power	[174]	images/W; fps/W	Smartphone & laptop AI-inference benchmarks
B CANDLE Power Study	[175]	J/epoch; GFLOPS/W	DOE accelerator procurement guidance
B LULESH/miniFE Energy	[176]	J/iteration	DVFS + autotuning baselines
B ExaSMR Power Benchmark	[177]	J/neutron; energy-vs-accuracy curve	Energy budget strategy in nuclear simulations
B EE-HPC-WG Energy Benchmark	[178]	draft node/job spec; JSON trace	Toward common HPC energy standard
B HPC-AI500 Energy Track	[179]	planned: GFLOPS/W; tokens/J	Mixed AI/HPC cluster evaluations
B PARSEC-3.1 Energy Extension	[180]	W; J via PAPI-RAPL; J/op; EDP	Pre-silicon DVFS research
B CosmoFlow-Power	[181]	J/epoch; GFLOPS/W	CNN scaling on 15 k+ GPUs
B HACC Energy Add-on	[182]	J/particle update	N-body cosmology power studies
B DeepCAM-Energy	[183]	J/epoch (UNet)	Climate-analytics accelerator studies
B OpenIFS-Energy	[184]	kWh/model-day; W timeline	Weather-model node comparison
B GROMACS-EE	[185]	J/ns; W/GPU	MD clock-vs-accuracy trade-offs
B NAMD-Power	[186]	Energy-Delay-Product (ApoA1)	Summit node DVFS optimisation
B QE Energy Suite	[187]	J/SCF step; GFLOPS/W	DFT GPU-offload studies
B VASP-Power Harness	[188]	W; kWh/MD step	Materials-science accelerator compare
B OpenFOAM-Energy	[189]	J/1k iterations	CFD partitioning & mesh tuning
B InSAR-AI Power Kit	[190]	J/satellite scene	Edge-to-cloud EO inference cost
B H3D-Energy	[191]	J/hydrology timestep	Hydrology model DVFS exploration
Tool			
T PTDaemon/SERT Energy	[192]	calibrated W; kWh (node)	Lab reproducibility; Lot 9 labels
T Scaphandre	[193]	W; kWh (process/node, Prometheus)	Slurm dashboards; power-cap feedback
T Kepler	[194]	W/pod; J/pod (eBPF)	Energy observability in K8s clusters
T CodeCarbon	[195]	kWh; kg CO_2e (process)	Rapid CO_2 estimation in pipelines
T CarbonTracker	[196]	measured + predicted kWh; CO_2e	Scheduling DL jobs in low-carbon hours
T PowerPACK/Mont-Blanc	[197]	W; J for MPI/OpenMP mini-apps	Network-topology & DVFS studies
T Cray PAT Energy Counters	[198]	J/function; avg W	Kernel hotspot hunting on Shasta
T IBM PowerAPI (pmlib)	[199]	kWh (job/process)	Energy-aware scheduling on Summit
T NVIDIA DCGM Energy	[200]	W; J (GPU) 1Hz; telemetry	GPU power-cap discovery; Green500
T Intel VTune Power	[201]	package W; J/function	Roofline-vs-energy tuning on Xeon
T Cloudmesh GPU	[202]	Power Draw; Temperature	Temperature and energy frequency traces

3) Leveraging Previous Work: As we can see from the table, a large number of tools and benchmarks exist, and we can leverage them to work towards a FAIR-based approach on energy benchmarks. This is all the more important when developing concise carpentry and democratization efforts. The distinction in the layered architecture for energy benchmarks also helps, as it is often not possible or desirable to address all layers at once. It is evident that energy benchmarking, in itself, is a complex research topic, and that carpentry efforts must be established to bring this knowledge forward and enhance AI benchmarks into AI energy benchmarks.

F. Simulation as a Tool to Benefit AI Benchmark Carpentry and Democratization

Simulating AI hardware and software infrastructures offers an opportunity to democratize AI benchmarking and impact AI development. This is especially useful for those (a) without direct access to the hardware on which the AI benchmarks run, and therefore can use simulations to estimate its behavior; and (b) planning large-scale experiments, who can use simulations to assess the impact on real hardware and infrastructure.

As part of this, recent work in the modeling and simulation community has significantly expanded users' options for studying how their ML workload optimizations affect them. Although there is a wide array of tools that can be used, we focus on four of the most popular, widely used tools: Accel-Sim [203], gem5 [204, 205], SST [206, 207], and Digital Twins [208]. These tools are often used in academia, industry, and national labs because they enable high-fidelity, early-stage design exploration. Moreover, they enable users who do not have access to real hardware or

are prototyping optimizations for hardware that does not yet exist to simulate the behavior of popular ML workloads while balancing performance and power trade-offs.

Accel-Sim [203]: For users interested in simulating ML workloads on modern NVIDIA (Volta through Blackwell) GPUs, Accel-Sim offers a great combination of high fidelity and usability. Accel-Sim builds upon the popular GPGPU-Sim [209], and has an integrated power model [210]. This allows users to examine power and performance tradeoffs for ML workloads.

Currently, Accel-Sim supports running ML workloads in three formats: (1) direct CUDA source code, (2) CUDA programs with library calls where the library includes the PTX for the library calls (only for CUDA 8.1 and earlier [211]), (3) and direct SASS (NVIDIA's machine assembly language) execution. As NVIDIA's libraries (e.g., cuDNN, cuBLAS) grow increasingly complex, and software like PyTorch add additional complexity on top of these libraries, the third option is the most popular as it can trace through multiple layers of software (e.g., PyTorch, cuBLAS). Moreover, to make the simulator's runtime more tractable, recent work has demonstrated how to identify and simulate a representative subset of a given workload without significantly compromising accuracy [212]. Thus, Accel-Sim is widely used by users who want to improve the efficiency of a given GPU. However, since Accel-Sim focuses on the GPU, it may not be best for users who want to study interactions with other system components (e.g., the CPU or other accelerators). Accel-Sim also does not heavily focus on the GPU cache coherence or memory consistency.

gem5 [204, 205]: The gem5 simulator is another popular tool used in computer system research to evaluate novel hardware designs. It provides a robust API for researchers to modify and extend current models and to create new models in the gem5 infrastructure. The gem5 simulator implements many models for system component including CPUs (out-of-order designs, in-order designs, and others), AMD and ARM GPUs [213], accelerators [214, 215, 216], various memories, on-chip interconnects, coherent caches, I/O devices, and many others. These gem5 models have enough fidelity to boot Linux, run unmodified workloads, and investigate cross-layer designs.

Thus, gem5 enables rapid prototyping of hardware-software co-designs across the computing stack. For example, users can prototype optimizations to the compiler, OS, or runtime in tandem with architectural changes and study the implications of their design choices. Like Accel-Sim, gem5 has an integrated power model [217] and also supports running popular ML workloads both natively and through frameworks like PyTorch – including adding support for advanced techniques to tradeoff simulation time for reduced fidelity in less important application regions [218, 219, 220]. However, gem5's support for ML workloads differs in three key ways from Accel-Sim's. First, unlike Accel-Sim, gem5's support for ML workloads spans across different types of compute devices, including CPUs and accelerators. Second, gem5 currently focuses its support on AMD GPUs. Since AMD's GPU runtime and drivers are open-source, this enables gem5 to model co-design between additional layers of the computing stack because it simulates all of those layers (unlike Accel-Sim). Third and finally, gem5 also has highly accurate models for cache coherence, memory consistency, and interfaces between components in the system like the GPU's Command Processor. Thus, gem5 may be a good choice for users wanting to study how ML workloads behave across system components or who want to prototype optimizations across layers of the computing stack. However, since many users focus on NVIDIA GPUs and gem5 currently does not support them, users deeply tied to NVIDIA's ecosystem will not find it useful.

SST [206, 207]: Accel-Sim and gem5 focus on modeling a single GPU (Accel-Sim, gem5) or a single system-on-a-chip (gem5). However, modern, large-scale computing systems frequently have hundreds or thousands of processors (e.g., GPUs) integrated together. Thus, the Structural Simulation Toolkit (SST) is a good option for users who want to study ML workloads in rack-scale systems. Instead of using high fidelity, but often slow models for components like processors (like Accel-Sim and gem5 do), SST utilizes analytical models for these components and focuses on modeling the network across many components, making it faster and scalable. However, for users who want to focus on both smaller- and larger-scale systems, both Accel-Sim [221, 222] and gem5 [223, 224] have integrated their models with SST – potentially providing the best of both worlds.

ExaDigiT [208]: To study AI workloads at datacenter or supercomputer scale, ExaDigiT provides a holistic digital twin framework that models the coupled behavior of workloads, compute, power, and cooling subsystems. Unlike simulators such as Accel-Sim, gem5, or SST, which operate at device- or node-level timescales, ExaDigiT enables large-scale modeling of system dynamics over operational timescales—capturing interactions that are difficult to observe or measure directly in production environments. This framework further provides a means to evaluate operational strategies, perform "what-if" analyses, and uncover complex, cross-disciplinary transient behaviors that emerge from the tight coupling of workloads, compute, power, and cooling.

ExaDigiT consists of three coupled modules: (1) a *resource allocator and power simulator (RAPS)* for replaying telemetry, simulating the scheduling of real or synthetic workloads, and dynamically estimating energy consumption; (2) a *thermo-fluid cooling module* for predicting pressures, temperatures, flow rates, system-level control responses, and overall power-usage effectiveness (PUE); and (3) a *visual analytics module* that integrates both a web-based dashboard and extended reality (XR) interfaces for immersive exploration of system behavior in augmented, virtual, or mixed reality.

Operating at coarser timescales than cycle-accurate simu-

TABLE VIII: Example Simulation Tools that Benefit AI Benchmark Simulations.

Tool/Software	Scale	Benefits	Application
Accel-Sim [203]	Single- and multi-**GPU**	High fidelity, usability, integrated power model, supports NVIDIA GPUs.	Examining power/performance tradeoffs; improving GPU efficiency.
gem5 [204, 205]	Single- and multi-**CPU, GPU, and system-on-a-chip**	High fidelity, hardware-software co-design, models cache coherence, interconnects, and memory consistency, supports accelerators and AMD/ARM GPUs.	Studying ML workload behavior across components; prototyping optimizations across layers.
SST [206, 207]	**Rack-scale** systems	Faster, scalable, models networking, utilizes analytical models.	Studying ML workload behavior in large-scale systems.
ExaDigiT [208]	**Datacenter- or supercomputer-level**	Models interactions among workloads, scheduling, power, networking, and cooling, including physical footprint.	Examining ML workload behavior at the largest scales.

lators, ExaDigiT enables comprehensive studies of power, cooling, and scheduling interactions across the full supercomputer. It has been applied to analyze how scheduling policies influence power and cooling dynamics [225], used as a reinforcement learning environment for training optimal scheduling agents [226], and to perform "virtual" benchmarking of large-scale LLM training workloads [227].
Summary: Collectively, these simulation frameworks span a continuum of modeling fidelity and scale—from device-level, cycle-accurate simulators such as Accel-Sim and gem5, to system- and datacenter-level models such as SST and ExaDigiT. By enabling controlled, repeatable, and cost-effective experimentation, they serve to democratize AI benchmarking in the design-space exploration of emerging architectures. As AI workloads continue to push the limits of power, cooling, and scheduling efficiency, such simulation-based tools will become indispensable for evaluating new ideas before committing to physical deployment.

V. Sharing Benchmarks

Beyond the creation of new benchmarks, *sharing* benchmakrs is an essential aspect of benchmark carpentry. To this end, integrating the FAIR principles is of paramount importance.
Benchmark sharing is best supported through hosting the code in a public repository that provides well-documented, executable workflows, thereby enabling others to reproduce the benchmark and compare results. Standard development practices, such as using Python Notebooks or scripts in other programming languages, as well as standard libraries, are recommended. More complex benchmarks may benefit from formal build processes (e.g., using makefiles) and dependency management through package managers. Containerization offers additional advantages, simplifying configuration and improving portability across environments.
To further support FAIRness, benchmark results should include standardized metadata, facilitating consistent comparison and analysis across studies.
While existing platforms such as Hugging Face and Kaggle provide mechanisms for sharing benchmarks, results, and leaderboards, fostering community capacity to host them independently remains valuable. Initiatives such as MLCommons illustrate how communities can maintain open, transparent benchmarking ecosystems. Educational efforts could be developed to train researchers and practitioners in these practices.
Finally, with the growing prominence of agentic AI, it is worth exploring its potential for automating the benchmarking lifecycle—including benchmark execution, result generation, and report synthesis. For example, the MLCommons Science Working Group is investigating how agentic AI can be applied to scientific benchmarks, particularly those involving time series analysis.

VI. Towards an AI Benchmark Carpentry Curriculum

Based on the lessons learned and our observations from domain experts, we have devised the following exemplary curriculum addressing AI benchmark carpentry.

- **Software Carpentry Foundational Tools and Practices:**
  Before addressing benchmark carpentry, we recommend that participants will review and learn about basic fundamental tools and practices. As they already exist as part of Software Carpentry, they can be reused. However, it may be of advantage to adapt certain aspects to explicitly utilize examples that focus on AI benchmarks and not just any arbitrary software carpentry project.
  - Programming Skills: Proficiency in Python, Jupyter Notebooks, focusing on reproducible coding practices, including documentation, and reproducibility.
  - Version Control: Git for tracking changes and collaboration.
  - Command-Line Proficiency: Unix shell for efficient data manipulation.

  - Data Management: Techniques for data cleaning, transformation, and visualization.
  - Learning from Online AI/LLM Resources: Leveraging large language models and online tutorials for benchmarking insights and guidance.
- **AI Benchmarking Fundamentals:**
  Having a basic understanding of AI Benchmarking is important for designing, evaluating, and improving AI systems. Benchmarks provide a standardized way to measure performance, compare models, and identify areas for optimization. By introducing benchmarking methodologies, examples, and metrics, participants gain the tools to critically assess AI models. Effective simple visualization practices help communicating results in a transparent, reproducible fashion related to real-world examples.
  - Benchmarking Methodologies: Introduction to frameworks such as MLPerf and AIBench.
  - Scenario-Based Benchmarks: Creating benchmarks that simulate real-world AI applications.
  - Performance Metrics: Throughput, latency, accuracy, and resource utilization.
  - Displaying Information with Graphs: Visualizing benchmark results for better analysis and interpretation.
- **Reproducibility and Experiment Management:**
  Especially for benchmarks, it is not only important to document the code, but to document the results so we enable reproducibility. This includes documenting workflows and data provenance in case prior work and data are integrated. Thus, applying the FAIR principles—making data and experiments Findable, Accessible, Interoperable, and Reusable—enhances transparency and promotes collaboration across teams and institutions.
  - Experiment Documentation: Importance of detailed documentation for reproducibility and adherence to FAIR principles.
  - Automated Workflows: Using Docker and CI/CD pipelines to automate benchmarking processes.
  - Data Provenance: Tracking data sources and transformations for transparency, traceability, and reuse.
  - FAIR: Apply the FAIR principle to AI benchmarks.
- **Ethical Considerations and Bias Mitigation:**
  It is important to address the ethical implications of conducting Benchmarks. Here, we not just focus on societal impacts, but also on the reporting of bias, fairness conducted potentially through hardware, software, and even vendor impacts.
  - Bias Detection: Methods to identify and mitigate biases in AI models and datasets.
  - Fairness Metrics: Metrics to assess and ensure fairness in AI systems.
  - Ethical Implications: Discussion on societal impacts and ethical decision-making.
- **Carpentry Principles in Practice:**
  A practical experience will be introduced to showcase the principles of AI benchmarking techniques. For this, a small, manageable datasets, and AI algorithm are used. The project may be conducted individually or in groups, while a walkthrough will also be available. An expansion to this AI-based benchmark will be the hosting and deployment of a leaderboard. Contributors can post their results in this shared leaderboard for the compute systems they have access to.
  - Hands-On Workshops: Practical sessions applying benchmarking techniques to real datasets.
  - Collaborative Projects: Group projects to foster teamwork and problem-solving skills.
  - Open-Source Contributions: Participation in community AI benchmarking initiatives.
- **Special Topics:**
  As we have seen from the previous section, several aspects have a great impact on AI benchmarking, which is so far not covered by other carpentry efforts. This includes energy benchmarking, simulation of hardware to estimate performance, and performance tuning with a focus on AI. Instead of just setting up a leaderboard through, for example, a Docker container, selected parties may have an interest in finding out more about setting up such leaderboards and hosting them.
  - Energy Efficiency: Measuring power consumption and optimizing AI workloads for lower energy usage.
  - Simulation: Using synthetic data and simulated environments for benchmarking when real data is limited.
  - Performance Tuning: Techniques for optimizing model execution, hardware utilization, and system throughput.
  - Leaderboard Management: Designing, maintaining, and validating AI benchmark leaderboards for reproducibility and fairness.
  - To provide users a starting point, presenting the community with a collection of benchmarks can be useful and has been spearheaded at [228].

From the extensive surveys and numerous examples it is important that to start one ought to begin with the most elementary efforts and grow them continuously. As such, we recommend adding specific lessons when we discover they need to be added by the community. Also, we must involve the community itself and allow for contributions of tutorials from a wide variety of groups.

VII. Towards AI Benchmark Democratizing

Our goal is to make AI benchmarking transparent, reproducible, and community-driven. Democratization empowers a broader range of participants to contribute to and learn from AI performance evaluation.

Introducing democratization tools, datasets, and evaluation frameworks that are openly accessible and easy to use can allow anyone—from students to independent researchers—to measure, compare, and improve AI models. One of the biggest hurdles we find is that some benchmarks, probably rightfully so, target hyperscale or leadership-class machines. However, in order to increase the community and raise awareness, smaller scale benchmarks need to be available.

As such, the following aspects can improve democratization:

- **Accessibility:**
  - Benchmarks, datasets, and tooling ought to be open-source or freely available.
  - Users may not need to rely on expensive hardware or proprietary software to participate.
  - Examples can be leveraged to develop new benchmarks. One can start with examples provided by MLCommons open datasets, pre-built benchmarking pipelines, and Jupyter notebooks with ready-to-run benchmarks.
- **Usability:**
  - Interfaces, documentation, and examples in existing efforts can serve as starting point to developing user-friendly, allowing non-experts to run benchmarks.
  - Providing automated scripts and tutorials reduces the barrier to entry.
- **Transparency:**
  - Specifying clear definitions of metrics, scoring methods, and evaluation procedures ensures everyone understands the results.
  - Improved transparency addresses the hide everything in a "black box" approach, where only insiders can interpret outcomes.
- **Community Participation:**
  - Anyone with minimal but sufficient knowledge should be able to contribute to benchmarks, improve tools, or submit models.
  - Democratization also means encouraging collaboration and reproducibility across institutions and geographies (e.g., engaging the broader community).
- **Impact:**
  - Through democratization, smaller teams or educational institutions can contribute and benefit from learning, competing, and comparing AI benchmarks.
  - Through democratization, fairness and innovation is fostered because knowledge and evaluation methods are disseminated.

VIII. Conclusion

Overall, this comprehensive paper has explored the motivations and pathways for creating a holistic benchmark carpentry effort, paying specific attention to aspects that can democratize AI benchmarks. This was achieved by (a) providing standardized and formal definitions of benchmarks, and (b) identifying a representative set of benchmarks related to AI activities. Finally, we propose an AI Benchmark Carpentry curriculum that integrates the various topics discussed into a structured learning activities to empower practitioners with reproducible coding practices, experiment-management skills, and an ethical lens on benchmarking. By embedding FAIR principles, bias-mitigation techniques, and performance-tuning modules, the curriculum offers a scalable pathway for communities—from academic labs to industry R&D—to build, share, and improve benchmarks in a collaborative, transparent manner.

Together, these activities foster democratization of AI benchmarks and can be utilized to grow the community and the understanding on how benchmarks may effect an individual activity or even community. While deploying such activities, we hope to grow community awareness and overcome the lack of well defined activities to educate the community in this regard. While fostering these activities we also address the need for more easily develop dynamic and adaptable benchmarks.

Acknowledgment

We have used at one point "ChatGPT" to improve upon the grammar of selected sections with the question: "Improve the grammar of ...". However, we stopped that practice early on due to wrong corrections, and have used Grammarly throughout the paper.

The work was in part sponsored by NSF Grant #2346173 and # 2303700, POSE: Phase II: MLCommons Research for Science: Enabling Open-Source Ecosystems for Scientific Foundation Models by Community Standards and Benchmarks

The portion of this work done at UW-Madison is supported in part by NSF grant CNS-2312688 and by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, under Award Number DE-SC-0026036.

This manuscript has been in part authored by FermiForward Discovery Group, LLC under Contract No. 89243024CSC000002 with the U.S. Department of Energy, Office of Science, Office of High Energy Physics. Fermilab Report Number FERMILAB-PUB-25-0835-CSAID.

This work was supported by DOE ASCR Microelectronics Science Research Center Projects, BIA. This material is based upon work supported by the U.S. Department of Energy, Office of Science, under contract number DE-AC02-06CH11357.

This research was in part sponsored in part by and used resources of the Oak Ridge Leadership Computing Facility (OLCF), which is a DOE Office of Science User Facility at the Oak Ridge National Laboratory (ORNL) supported by the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

Shirley Moore's work on this paper was supported by the Department of Energy Office of Science under award #DE-

SC0024352.
Kirkpatrick's work was made possible through the National Science Foundation award #2226453.
This research was funded in part by and used resources at the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357.
Research was sponsored by the Department of the Air Force Artificial Intelligence Accelerator and was accomplished under Cooperative Agreement Number FA8750-19-2-1000. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Department of the Air Force or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.
Gary Mazzaferro has participated in discussions as part of the working group meetings surrounding benchmark definitions and applicability.

References

[1] DOE, *Trillion parameter consortium*, [Online; accessed 2025-11-30], Aug. 2023. [Online]. Available: https://tpc.dev/.

[2] R. Stevens, "Argonne's "AuroraGPT" Project," *Trillion Parameter Consortium Seminar*, TPC, 2023.

[3] Wikipedia, *Benchmark (computing)*, [Online; accessed 2025-09-23], Jun. 2005. [Online]. Available: https://en.wikipedia.org/wiki/Benchmark_%28computing%29.

[4] J. J. Dongarra, "Performance of various computers using standard linear equations software," University of Tennessee, Knoxville / Oak Ridge National Laboratory, Tech. Rep. Technical Report CS-89-85, 1989. [Online]. Available: http://www.netlib.org/benchmark/performance.ps.

[5] J. J. Dongarra, M. A. Heroux, and P. Luszczek, "High-performance conjugate-gradient benchmark: A new metric for ranking high-performance computing systems," *International Journal of High Performance Computing Applications*, vol. 30, no. 1, pp. 3–8, 2016. DOI: 10.1177/1094342015593158. [Online]. Available: https://doi.org/10.1177/1094342015593158.

[6] Google Cloud Platform and contributors, *Perfkitbenchmarker*, GitHub, 2025. [Online]. Available: https://github.com/GoogleCloudPlatform/PerfKitBenchmarker.

[7] I. S. Committee, *Io500: A benchmarking suite for hpc storage i/o performance*, Web Page, 2025. [Online]. Available: https://io500.org.

[8] G. Wilson, "Software carpentry: Lessons learned," *F1000Research*, vol. 3, 2014. DOI: 10.12688/f1000research.3-62.v2. [Online]. Available: https://doi.org/10.12688/f1000research.3-62.v2.

[9] Software Carpentry, *Software carpentry*, https://software-carpentry.org/, Accessed: 2025-05-28, 2024.

[10] The Carpentries, *Data carpentry*, https://datacarpentry.org, Accessed: 2025-10-23, 2025.

[11] A. Reid, T. Keller, A. O'Cais, A. A. Rasel, W. Purwanto, J. Herriman, B. Muite, and M.-A. Hermanns, "Hpc carpentry: Recent progress and incubation toward an official carpentries lesson program," *Journal of Computational Science*, vol. 16, no. 1, 2025.

[12] The Carpentries / HPC Carpentry community, *Hpc carpentry*, https://hpc-carpentry.org, Accessed: 2025-10-23, 2025.

[13] G. von Laszewski, J. P. Fleischer, R. Knuuti, G. C. Fox, J. Kolessar, T. S. Butler, and J. Fox, "Opportunities for enhancing mlcommons efforts while leveraging insights from educational mlcommons earthquake benchmarks efforts," *Frontiers in High Performance Computing*, vol. 1, no. 1233877, p. 31, Oct. 2023, ISSN: 2813-7337. DOI: 10.3389/fhpcp.2023.1233877. [Online]. Available: https://www.frontiersin.org/journals/high-performance-computing/articles/10.3389/fhpcp.2023.1233877.

[14] J. J. Dongarra, P. Luszczek, and A. Petitet, "The LINPACK benchmark: Past, present, and future," *Concurrency and Computation: Practice and Experience*, vol. 15, no. 9, pp. 803–820, Aug. 2003, ISSN 1532-0634. DOI: 10.1002/cpe.728.

[15] J. J. Dongarra, J. Bunch, C. Moler, and G. W. Stewart, *LINPACK User's Guide*. Philadelphia, PA, USA: Society of Industrial and Applied Mathematics, 1979.

[16] W.-C. Feng, R. Ge, and K. W. Cameron, "Power and energy profiling of scientific applications on distributed systems," in *19th IEEE International Parallel and Distributed Processing Symposium (IPDPS 05)*, Denver, CO, USA, 2005.

[17] K. W. Cameron, R. Ge, and X. Feng, "High-performance, power-aware, distributed computing for scientific applications," *IEEE Computer*, vol. 38, no. 11, pp. 40–47, 2005.

[18] SPEC, *The SPEC power benchmark*, 2008. [Online]. Available: www.spec.org/power_ssj2008/.

[19] *Specpower_ssj2008*, https://spec.org/power_ssj2008/, Watts per transaction and operations per Watt for enterprise servers, 2025. (visited on 05/06/2025).

[20] W.-c. Feng and K. Cameron, "The green500 list: Encouraging sustainable supercomputing," *Computer*, vol. 40, no. 12, pp. 50–55, Dec. 2007, ISSN: 0018-9162. DOI: 10.1109/MC.2007.445. [Online]. Available: https://doi.org/10.1109/MC.2007.445.

[21] A. Abdelfattah, H. Anzt, E. G. Boman, E. Carson, T. Cojean, J. Dongarra, A. Fox, M. Gates, N. J. Higham, X. S. Li, J. Loe, P. Luszczek, S. Pranesh, S. Rajamanickam, T. Ribizel, B. F. Smith, K. Swirydowicz, S. Thomas, S. Tomov, Y. M. Tsai, and

U. M. Yang, "A survey of numerical linear algebra methods utilizing mixed-precision arithmetic," *The International Journal of High Performance Computing Applications*, vol. 35, no. 4, pp. 344–369, 2021. DOI: 10.1177/10943420211003313. eprint: https://doi.org/10.1177/10943420211003313. [Online]. Available: https://doi.org/10.1177/10943420211003313.

[22] Cornell University, *Arxiv.org e-print archive*, [Online; accessed 2025-10-01], Oct. 2025. [Online]. Available: https://arxiv.org/.

[23] *Google scholar*, [Online; accessed 2025-10-01], Oct. 2025. [Online]. Available: https://scholar.google.com/.

[24] G. von Laszewski, B. Hawks, M. Colombo, R. Shiraishi, A. Krishnan, N. Tran, and G. C. Fox, *Mlcommons science working group ai benchmarks collection*, GitHub, Online Collection: =https://mlcommons-science.github.io/benchmark/, Jun. 2025. [Online]. Available: https://mlcommons-science.github.io/benchmark/benchmarks.pdf.

[25] B. Hawks, G. von Laszewski, M. D. Sinclair, M. Colombo, S. Venkataraman, R. Jain, Y. Jiang, N. Tran, and G. Fox, *An MLCommons Scientific Benchmarks Ontology*, arXiv, 2025. [Online]. Available: https://arxiv.org/abs/2511.05614.

[26] "Machine learning innovation to benefit everyone," *Web page*, Apr. 2023, https://mlcommons.org/ [Accessed April 13, 2023]. [Online]. Available: https://mlcommons.org/.

[27] G. von Laszewski, N. Tran, *et al.*, *Mlcommons-science/benchmark*, [Online; accessed 2025-10-01], Oct. 2025. [Online]. Available: https://github.com/mlcommons-science/benchmark.

[28] T. Nguyen, J. Jewik, H. Bansal, P. Sharma, and A. Grover, *Climatelearn: Benchmarking machine learning for weather and climate modeling*, arXiv, 2023. eprint: 2307.01909 (cs.LG). [Online]. Available: https://arxiv.org/abs/2307.01909.

[29] J. Thiyagalingam, G. von Laszewski, J. Yin, M. Emani, J. Papay, G. Barrett, P. Luszczek, A. Tsaris, C. Kirkpatrick, F. Wang, T. Gibbs, V. Vishwanath, M. Shankar, G. Fox, and T. Hey, "Ai benchmarking for science: Efforts from the mlcommons science working group," in *High Performance Computing. ISC High Performance 2022 International Workshops*, H. Anzt, A. Bienz, P. Luszczek, and M. Baboulin, Eds., Cham: Springer International Publishing, 2022, pp. 47–64, ISBN: 978-3-031-23220-6.

[30] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord, *Think you have solved question answering? try arc, the ai2 reasoning challenge*, arXiv:1803.05457v1, 2018. DOI: 10.48550/arXiv.1803.05457.

[31] Y. Fang, N. Zhang, Z. Chen, L. Guo, X. Fan, and H. Chen, *Domain-agnostic molecular generation with chemical feedback*, arXiv, 2024. eprint: 2301.11259 (cs.LG). [Online]. Available: https://arxiv.org/abs/2301.11259.

[32] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec, *Open Graph Benchmark: Datasets for Machine Learning on Graphs*, arXiv, 2021. eprint: 2005.00687 (cs.LG). [Online]. Available: https://arxiv.org/abs/2005.00687.

[33] H. Zhang, J. Sun, R. Chen, W. Liu, Z. Yuan, X. Zheng, Z. Wang, Z. Yang, H. Yan, H.-S. Zhong, X. Wang, W. Ouyang, F. Yang, and N. Dong, "Empowering and assessing the utility of large language models in crop science," in *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. [Online]. Available: https://openreview.net/forum?id=hMj6jZ6JWU.

[34] M. Tian, L. Gao, S. D. Zhang, X. Chen, C. Fan, X. Guo, R. Haas, P. Ji, K. Krongchon, Y. Li, S. Liu, D. Luo, Y. Ma, H. Tong, K. Trinh, C. Tian, Z. Wang, B. Wu, Y. Xiong, S. Yin, M. Zhu, K. Lieret, Y. Lu, G. Liu, Y. Du, T. Tao, O. Press, J. Callan, E. Huerta, and H. Peng, *Scicode: A research coding benchmark curated by scientists*, arXiv, 2024. eprint: 2407.13168 (cs.AI). [Online]. Available: https://arxiv.org/abs/2407.13168.

[35] C. Krause *et al.*, *Calochallenge 2022: A community challenge for fast calorimeter simulation*, arXiv, 2024. eprint: 2410.21611 (physics.ins-det). [Online]. Available: https://arxiv.org/abs/2410.21611.

[36] M. Takamoto, T. Praditia, R. Leiteritz, D. MacKinlay, F. Alesiani, D. Pflüger, and M. Niepert, *Pdebench: An extensive benchmark for scientific machine learning*, arXiv, 2024. eprint: 2210.07182 (cs.LG). [Online]. Available: https://arxiv.org/abs/2210.07182.

[37] Y. Wang, T. Wang, Y. Zhang, H. Zhang, H. Zheng, G. Zheng, and L. Kong, "Urbandatalayer: A unified data pipeline for urban science," in *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, Eds., vol. 37, Curran Associates, Inc., 2024, pp. 7296–7310. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2024/file/0db7f135f6991e8cec5e516ecc66bfba-Paper-Datasets_and_Benchmarks_Track.pdf.

[38] S. Pramanick, R. Chellappa, and S. Venugopalan, *Spiqa: A dataset for multimodal question answering on scientific papers*, arXiv, 2025. eprint: 2407.09413 (cs.CL). [Online]. Available: https://arxiv.org/abs/2407.09413.

[39] A. Karargyris *et al.*, "Federated benchmarking of medical artificial intelligence with medperf," *Nature*

Machine Intelligence, vol. 5, no. 7, pp. 799–810, Jul. 2023. DOI: 10.1038/s42256-023-00652-2. [Online]. Available: https://doi.org/10.1038/s42256-023-00652-2.

[40] K. X. Nguyen, F. Qiao, A. Trembanis, and X. Peng, *Seafloorai: A large-scale vision-language dataset for seafloor geological survey*, arXiv, 2024. eprint: 2411.00172 (cs.CV). [Online]. Available: https://arxiv.org/abs/2411.00172.

[41] D. Zou, S. Liu, S. Miao, V. Fung, S. Chang, and P. Li, "Gess: Benchmarking geometric deep learning under scientific applications with distribution shifts," in *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, Eds., vol. 37, Curran Associates, Inc., 2024, pp. 92 499–92 528. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2024/file/a8063075b00168dc39bc81683619f1a8-Paper-Datasets_and_Benchmarks_Track.pdf.

[42] L. Chanussot, A. Das, S. Goyal, T. Lavril, M. Shuaibi, M. Riviere, K. Tran, J. Heras-Domingo, C. Ho, W. Hu, A. Palizhati, A. Sriram, B. Wood, J. Yoon, D. Parikh, C. L. Zitnick, and Z. Ulissi, "The open catalyst 2020 (oc20) dataset and community challenges," *ACS Catalysis*, vol. 11, no. 10, pp. 6059–6072, 2021. DOI: 10.1021/acscatal.0c04525. [Online]. Available: https://pubs.acs.org/doi/10.1021/acscatal.0c04525.

[43] R. Tran, J. Lan, M. Shuaibi, B. M. Wood, S. Goyal, A. Das, J. Heras-Domingo, A. Kolluru, A. Rizvi, N. Shoghi, A. Sriram, F. Therrien, J. Abed, O. Voznyy, E. H. Sargent, Z. Ulissi, and C. L. Zitnick, "The open catalyst 2022 (oc22) dataset and challenges for oxide electrocatalysts," *ACS Catalysis*, vol. 13, no. 5, pp. 3066–3084, 2023. DOI: 10.1021/acscatal.2c05426. [Online]. Available: https://pubs.acs.org/doi/10.1021/acscatal.2c05426.

[44] J. Duarte, N. Tran, B. Hawks, C. Herwig, J. Muhizi, S. Prakash, and V. J. Reddi, *Fastml science benchmarks: Accelerating real-time scientific edge machine learning*, arXiv, 2022. eprint: 2207.07958 (cs.LG). [Online]. Available: https://arxiv.org/abs/2207.07958.

[45] S. Farrell *et al.*, *Mlperf hpc: A holistic benchmark suite for scientific machine learning on hpc systems*, arXiv, 2021. eprint: 2110.11466 (cs.LG). [Online]. Available: https://arxiv.org/abs/2110.11466.

[46] E. G. Campolongo *et al.*, *Building machine learning challenges for anomaly detection in science*, arXiv, 2025. eprint: 2503.02112 (cs.LG). [Online]. Available: https://arxiv.org/abs/2503.02112.

[47] P. Chen, L. Peng, R. Jiao, Q. Mo, Z. Wang, W. Huang, Y. Liu, and Y. Lu, "Learning superconductivity from ordered and disordered material structures," in *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, Eds., vol. 37, Curran Associates, Inc., 2024, pp. 108 902–108 928. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2024/file/c4e3b55ed4ac9ba52d7df11f8bddbbf4-Paper-Datasets_and_Benchmarks_Track.pdf.

[48] E. Luo, J. Jia, Y. Xiong, X. Li, X. Guo, B. Yu, L. Wei, and X. Zhang, *Benchmarking ai scientists in omics data-driven biological research*, arXiv, 2025. eprint: 2505.08341 (cs.AI). [Online]. Available: https://arxiv.org/abs/2505.08341.

[49] R. Ohana, M. McCabe, L. Meyer, R. Morel, F. J. Agocs, M. Beneitez, M. Berger, B. Burkhart, S. B. Dalziel, D. B. Fielding, D. Fortunato, J. A. Goldberg, K. Hirashima, Y.-F. Jiang, R. R. Kerswell, S. Maddu, J. Miller, P. Mukhopadhyay, S. S. Nixon, J. Shen, R. Watteaux, B. R.-S. Blancard, F. Rozet, L. H. Parker, M. Cranmer, and S. Ho, "The well: A large-scale collection of diverse physics simulations for machine learning," in *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, Eds., vol. 37, Curran Associates, Inc., 2024, pp. 44 989–45 037. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2024/file/4f9a5acd91ac76569f2fe291b1f4772b-Paper-Datasets_and_Benchmarks_Track.pdf.

[50] D. Hendrycks, C. Burns, and S. Kadavath, *Measuring Massive Multitask Language Understanding*, 2021. [Online]. Available: https://arxiv.org/abs/2009.03300.

[51] J. Roberts, K. Han, and S. Albanie, "Satin: A multi-task metadataset for classifying satellite imagery using vision-language models," *ICCV Workshop: Towards the Next Generation of Computer Vision Datasets*, Mar. 2023. DOI: 10.48550/arXiv.2304.11619.

[52] D. Rein, B. L. Hou, and A. C. Stickland, *Gpqa: A graduate-level google-proof q and a benchmark*, 2023. [Online]. Available: https://arxiv.org/abs/2311.12022.

[53] H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, and K. Cobbe, "Let's verify step by step," *arXiv preprint arXiv:2305.20050*, 2023. DOI: 10.48550/arXiv.2305.20050. eprint: arXiv:2305.20050.

[54] N. Mudur, H. Cui, S. Venugopalan, P. Raccuglia, M. P. Brenner, and P. Norgaard, *Feabench: Evaluating language models on multiphysics reasoning ability*, arXiv, 2025. eprint: 2504.06260 (cs.AI). [Online]. Available: https://arxiv.org/abs/2504.06260.

[55] J. Weitz, D. Demler, L. McDermott, N. Tran, and J. Duarte, *Neural architecture codesign for fast physics applications*, arXiv, 2025. eprint: 2501.05515 (cs.LG).

[Online]. Available: https://arxiv.org/abs/2501.05515.

[56] K. Khrabrov, A. Ber, A. Tsypin, K. Ushenin, E. Rumiantsev, A. Telepov, D. Protasov, I. Shenbin, A. Alekseev, M. Shirokikh, S. Nikolenko, E. Tutubalina, and A. Kadurin, *Delta-squared dft: A universal quantum chemistry dataset of drug-like molecules and a benchmark for neural network potentials*, arXiv, 2024. eprint: 2406.14347 (physics.chem-ph). [Online]. Available: https://arxiv.org/abs/2406.14347.

[57] R. E. Peterson, A. Tanelus, C. Ick, B. Mimica, N. Francis, V. J. Ivan, A. Choudhri, A. L. Falkner, M. Murthy, D. M. Schneider, D. H. Sanes, and A. H. Williams, "Vocal call locator benchmark (vcl) for localizing rodent vocalizations from multi-channel audio," in *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, Eds., vol. 37, Curran Associates, Inc., 2024, pp. 106 370–106 382. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2024/file/c00d37d6b04d73b870b963a4d70051c1-Paper-Datasets_and_Benchmarks_Track.pdf.

[58] R. Bushuiev, A. Bushuiev, N. F. de Jonge, A. Young, F. Kretschmer, R. Samusevich, J. Heirman, F. Wang, L. Zhang, K. Dührkop, M. Ludwig, N. A. Haupt, A. Kalia, C. Brungs, R. Schmid, R. Greiner, B. Wang, D. S. Wishart, L.-P. Liu, J. Rousu, W. Bittremieux, H. Rost, T. D. Mak, S. Hassoun, F. Huber, J. J. van der Hooft, M. A. Stravs, S. Böcker, J. Sivic, and T. Pluskal, "Massspecgym: A benchmark for the discovery and identification of molecules," in *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, Eds., vol. 37, Curran Associates, Inc., 2024, pp. 110 010–110 027. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2024/file/c6c31413d5c53b7d1c343c1498734b0f-Paper-Datasets_and_Benchmarks_Track.pdf.

[59] X. Zhong, Y. Gao, and S. Gururangan, *Spiqa: Scientific paper image question answering*, 2024. [Online]. Available: https://arxiv.org/abs/2407.09413.

[60] D. Rein, B. L. Hou, A. C. Stickland, J. Petty, R. Y. Pang, J. Dirani, J. Michael, and S. R. Bowman, *Gpqa: A graduate-level google-proof q and a benchmark*, arXiv, 2023. eprint: 2311.12022 (cs.AI). [Online]. Available: https://arxiv.org/abs/2311.12022.

[61] D. Jin, E. Pan, N. Oufattole, W.-H. Weng, H. Fang, and P. Szolovits, *What disease does this patient have? a large-scale open domain question answering dataset from medical exams*, arXiv, 2020. eprint: 2009.13081 (cs.CL). [Online]. Available: https://arxiv.org/abs/2009.13081.

[62] G. D. Guglielmo, B. Du, J. Campos, A. Boltasseva, A. V. Dixit, F. Fahim, Z. Kudyshev, S. Lopez, R. Ma, G. N. Perdue, N. Tran, O. Yesilyurt, and D. Bowring, *End-to-end workflow for machine learning-based qubit readout with qick and hls4ml*, arXiv, 2025. eprint: 2501.14663 (quant-ph). [Online]. Available: https://arxiv.org/abs/2501.14663.

[63] Y. Luo, Y. Chen, and Z. Zhang, *Cfdbench: A large-scale benchmark for machine learning methods in fluid dynamics*, 2024. [Online]. Available: https://arxiv.org/abs/2310.05963.

[64] H. Cui *et al.*, *Curie: Evaluating llms on multitask scientific long context understanding and reasoning*, arXiv, 2025. eprint: 2503.13517 (cs.CL). [Online]. Available: https://arxiv.org/abs/2503.13517.

[65] B. Parpillon, C. Syal, J. Yoo, J. Dickinson, M. Swartz, G. D. Guglielmo, A. Bean, D. Berry, M. B. Valentin, K. DiPetrillo, A. Badea, L. Gray, P. Maksimovic, C. Mills, M. S. Neubauer, G. Pradhan, N. Tran, D. Wen, and F. Fahim, *Smart pixels: In-pixel ai for on-sensor data filtering*, arXiv, 2024. eprint: 2406.14860 (physics.ins-det). [Online]. Available: https://arxiv.org/abs/2406.14860.

[66] T. Aarrestad, E. Govorkova, J. Ngadiuba, E. Puljak, M. Pierini, and K. A. Wozniak, *Unsupervised New Physics detection at 40 MHz: Training Dataset*, 2021. DOI: 10.5281/ZENODO.5046389. [Online]. Available: https://zenodo.org/record/5046389.

[67] J. Bowles, S. Ahmed, and M. Schuld, *Better than classical? the subtle art of benchmarking quantum machine learning models*, arXiv, 2024. [Online]. Available: https://arxiv.org/abs/2403.07059.

[68] P. Odagiu, Z. Que, J. Duarte, J. Haller, G. Kasieczka, A. Lobanov, V. Loncar, W. Luk, J. Ngadiuba, M. Pierini, P. Rincke, A. Seksaria, S. Summers, A. Sznajder, A. Tapper, and T. K. Aarrestad, *Ultrafast jet classification on fpgas for the hl-lhc*, arXiv, 2024. DOI: https://doi.org/10.1088/2632-2153/ad5f10. eprint: 2402.01876 (hep-ex). [Online]. Available: https://arxiv.org/abs/2402.01876.

[69] Z. Liu, H. Sharma, J.-S. Park, P. Kenesei, A. Miceli, J. Almer, R. Kettimuthu, and I. Foster, *Braggnn: Fast x-ray bragg peak analysis using deep learning*, arXiv, 2021. eprint: 2008.08198 (eess.IV). [Online]. Available: https://arxiv.org/abs/2008.08198.

[70] S. Qin, J. Agar, and N. Tran, "Extremely noisy 4d-tem strain mapping using cycle consistent spatial transforming autoencoders," in *AI for Accelerated Materials Design - NeurIPS 2023 Workshop*, 2023. [Online]. Available: https://openreview.net/forum?id=7yt3N0o0W9.

[71] D. Kafkes and J. S. John, *Boostr: A dataset for accelerator control systems*, arXiv, 2021. eprint: 2101.

08359 (physics.acc-ph). [Online]. Available: https://arxiv.org/abs/2101.08359.

[72] J. Kvapil *et al.*, *Intelligent experiments through real-time ai: Fast data processing and autonomous detector control for sphenix and future eic detectors*, arXiv, 2025. eprint: 2501.04845 (physics.ins-det). [Online]. Available: https://arxiv.org/abs/2501.04845.

[73] A. A. Abud *et al.*, *Deep Underground Neutrino Experiment (DUNE) Near Detector Conceptual Design Report*, arXiv, 2021. [Online]. Available: https://arxiv.org/abs/2103.13910.

[74] E. Glazer, E. Erdil, T. Besiroglu, D. Chicharro, E. Chen, A. Gunning, C. F. Olsson, J.-S. Denain, A. Ho, E. de Oliveira Santos, O. Järviniemi, M. Barnett, R. Sandler, M. Vrzala, J. Sevilla, Q. Ren, E. Pratt, L. Levine, G. Barkley, N. Stewart, B. Grechuk, T. Grechuk, S. V. Enugandla, and M. Wildon, *FrontierMath: A Benchmark for Evaluating Advanced Mathematical Reasoning in AI*, arXiv, 2024. eprint: 2411.04872 (cs.AI). [Online]. Available: https://arxiv.org/abs/2411.04872.

[75] vals.ai, *Public Enterprise LLM Benchmarks: AIME*, [Online accessed 2025-06-24], Mar. 2025. [Online]. Available: https://www.vals.ai/benchmarks/aime.

[76] M. Khan, S. Krave, V. Marinozzi, J. Ngadiuba, S. Stoynev, and N. Tran, "Benchmarking and interpreting real time quench detection algorithms," in *Fast Machine Learning for Science Conference 2024*, Purdue University, IN: indico.cern.ch, Oct. 2024. [Online]. Available: https://indico.cern.ch/event/1387540/contributions/6153618/attachments/2948441/5182077/fast_ml_magnets_2024_final.pdf.

[77] A. Jain, S. P. Ong, G. Hautier, W. Chen, W. D. Richards, S. Dacek, S. Cholia, D. Gunter, D. Skinner, G. Ceder, and K. A. Persson, "The Materials Project: A Materials Genome Approach," *APL Materials*, vol. 1, no. 1, 2013. DOI: 10.1063/1.4812323. [Online]. Available: https://materialsproject.org/.

[78] Y. Wei, R. F. Forelli, C. Hansen, J. P. Levesque, N. Tran, J. C. Agar, G. D. Guglielmo, M. E. Mauel, and G. A. Navratil, *Low latency optical-based mode tracking with machine learning deployed on fpgas on a tokamak*, arXiv, 2024. DOI: https://doi.org/10.1063/5.0190354. eprint: 2312.00128 (physics.plasm-ph). [Online]. Available: https://arxiv.org/abs/2312.00128.

[79] G. von Laszewski, J. Fleischer, G. C. Fox, J. Papay, S. Jackson, and J. Thiyagalingam, "Templated hybrid reusable computational analytics workflow management with cloudmesh, applied to the deep learning mlcommons cloudmask application," in *2023 IEEE 19th International Conference on e-Science (e-Science)*, 2023, pp. 1–6. DOI: 10.1109/e-Science58273.2023.10254942.

[80] M. Abadi *et al.*, "TensorFlow: A System for Large-Scale Machine Learning," *OSDI*, vol. 16, pp. 265–283, 2016, TensorBoard Profiler is a component of the TensorFlow ecosystem.

[81] A. Paszke *et al.*, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," *Advances in Neural Information Processing Systems*, vol. 32, 2019, PyTorch Profiler is part of the PyTorch library.

[82] Google, *Jax profiler*, Accessed: 2025-10-22, 2025.

[83] NVIDIA, *Nvidia deep learning profiler (dlprof)*, Accessed: 2025-10-22, 2025.

[84] NVIDIA Corporation, *NVIDIA Nsight Systems Documentation*, https://developer.nvidia.com/nsight-systems, Accessed: [Current Date], 2024.

[85] NVIDIA Corporation, *NVIDIA Nsight Compute Documentation*, https://docs.nvidia.com/nsight-compute/NsightCompute/index.html, Accessed: [Current Date], 2024.

[86] NVIDIA Corporation, *NVIDIA CUDA Profiler User's Guide*, Current Edition Number, e.g., 12.0, Documentation for nvprof. Available at https://docs.nvidia.com/cuda/profiler-users-guide/index.html, NVIDIA Corporation, Current Year, e.g., 2024.

[87] Intel Corporation, *Intel VTune Profiler Documentation*, https://www.intel.com/content/www/us/en/develop/documentation/vtune-help/, Accessed: [Current Date], 2024.

[88] AMD, *ROCm Documentation*, https://rocm.docs.amd.com/en/latest/, Accessed: [Current Date], 2024.

[89] J. Mellor-Crummey *et al.*, "HPCToolkit: Tools for Performance Analysis of Optimized Parallel Programs," *Concurrency and Computation: Practice and Experience*, vol. 24, no. 6, pp. 680–708, 2012.

[90] S. Shende and A. D. Malony, "The TAU Parallel Performance System," in *International Conference on Parallel and Distributed Computing and Systems*, 1998, pp. 489–493.

[91] G. Poulakos *et al.*, "Perfetto: A System-Wide Tracing Tool for Modern Applications," in *Proceedings of the 35th International Conference on Software Engineering (ICSE 2023)*, Based on the Perfetto project's scope for system tracing., 2023.

[92] S. Browne, J. J. Dongarra, N. Garner, K. S. London, and P. Mucci, "A Scalable Cross-Platform Infrastructure for Application Performance Tuning Using Hardware Counters," in *Proceedings Supercomputing 2000, November 4-10, 2000, Dallas, Texas, USA. IEEE Computer Society, CD-ROM*, J. Donnelley, Ed., IEEE Computer Society, 2000, p. 42. DOI: 10.1109/SC.2000.10029. [Online]. Available: https://doi.org/10.1109/SC.2000.10029.

[93] D. Wang *et al.*, "XLA: Optimizing Compiler for TensorFlow," in *Proceedings of the First Workshop*

on Systems for ML, XLA Profiler is a feature of the XLA compiler., 2017.

[94] M. (Facebook), *Torchdynamo & torchinductor debug tools*, Accessed: 2025-10-22, 2025.

[95] OpenAI, *Triton: An Intermediate Language for Tiled Tensor Computations*, https://openai.com/research/triton, The Triton profiler is part of the Triton compiler and language., 2019.

[96] NVIDIA Corporation, *NVIDIA Collective Communications Library (NCCL) Documentation*, https://docs.nvidia.com/deeplearning/nccl/user-guide/docs/index.html, Accessed: [Current Date], 2024.

[97] A. W. Services, *Aws sagemaker debugger*, Accessed: 2025-10-22, 2025.

[98] Microsoft, *Azure machine learning profiler*, Accessed: 2025-10-22, 2025.

[99] W. bibinitperiod Biases, *Weights & biases*, Accessed: 2025-10-22, 2025.

[100] C. ML, *Comet ml*, Accessed: 2025-10-22, 2025.

[101] Databricks, *Mlflow*, Accessed: 2025-10-22, 2025.

[102] M. /. Google, *Torch/tensorflow memory tools*, Accessed: 2025-10-22, 2025.

[103] P. S. Foundation, *Python profilers (cprofile, py-spy)*, Accessed: 2025-10-22, 2025.

[104] S. Che, B. M. Beckmann, S. K. Reinhardt, and K. Skadron, "Pannotia: Understanding Irregular GPGPU Graph Applications," in *IEEE International Symposium on Workload Characterization*, ser. IISWC, Sep. 2013, pp. 185–195. DOI: 10.1109/IISWC.2013.6704684.

[105] Y. Wang, Y. Pan, A. Davidson, Y. Wu, C. Yang, L. Wang, M. Osama, C. Yuan, W. Liu, A. T. Riffel, and J. D. Owens, "Gunrock: GPU Graph Analytics," *ACM Trans. Parallel Comput.*, vol. 4, no. 1, Aug. 2017, ISSN: 2329-4949. DOI: 10.1145/3108140. [Online]. Available: https://doi.org/10.1145/3108140.

[106] Lawrence Livermore National Labs, *CORAL-2 Benchmarks*, https://asc.llnl.gov/coral-2-benchmarks, 2020.

[107] Oak Ridge National Labs, *OLCF-6 Benchmarks*, https://www.olcf.ornl.gov/draft-olcf-6-technical-requirements/benchmarks/, 2024.

[108] J. Kim, A. D. Baczewski, T. D. Beaudet, A. Benali, M. C. Bennett, M. A. Berrill, N. S. Blunt, E. J. L. Borda, M. Casula, D. M. Ceperley, *et al.*, "QMCPACK: An Open Source ab initio Quantum Monte Carlo Package for the Electronic Structure of Atoms, Molecules and Solids," *Journal of Physics: Condensed Matter*, vol. 30, no. 19, p. 195 901, 2018.

[109] X. Wu, V. Taylor, J. M. Wozniak, R. Stevens, T. Brettin, and F. Xia, "Performance, Energy, and Scalability Analysis and Improvement of Parallel Cancer Deep Learning CANDLE Benchmarks," in *Proceedings of the 48th International Conference on Parallel Processing*, ser. ICPP, Kyoto, Japan: Association for Computing Machinery, 2019, ISBN: 9781450362955. DOI: 10.1145/3337821.3337905. [Online]. Available: https://doi.org/10.1145/3337821.3337905.

[110] C. R. Banbury, V. J. Reddi, P. Torelli, J. Holleman, N. Jeffries, C. Király, P. Montino, D. Kanter, S. Ahmed, D. Pau, U. Thakker, A. Torrini, P. Warden, J. Cordaro, G. D. Guglielmo, J. M. Duarte, S. Gibellini, V. Parekh, H. Tran, N. Tran, W. Niu, and X. Xu, "Mlperf tiny benchmark," *CoRR*, vol. abs/2106.07597, 2021. eprint: 2106.07597. [Online]. Available: https://arxiv.org/abs/2106.07597.

[111] T. Baruah, K. Shivdikar, S. Dong, Y. Sun, S. A. Mojumder, K. Jung, J. L. Abellán, Y. Ukidave, A. Joshi, J. Kim, and D. Kaeli, "GNNMark: A Benchmark Suite to Characterize Graph Neural Network Training on GPUs," in *IEEE International Symposium on Performance Analysis of Systems and Software*, ser. ISPASS, 2021, pp. 13–23. DOI: 10.1109/ISPASS51385.2021.00013.

[112] S. Dong and D. Kaeli, "DNNMark: A Deep Neural Network Benchmark Suite for GPUs," in *Proceedings of the General Purpose GPUs*, ser. GPGPU, Austin, TX, USA: ACM, 2017, pp. 63–72, ISBN: 978-1-4503-4915-4. DOI: 10.1145/3038228.3038239. [Online]. Available: http://doi.acm.org/10.1145/3038228.3038239.

[113] S. Narang and G. Diamos, *An update to DeepBench with a focus on deep learning inference*, https://svail.github.io/DeepBench-update/, 2017.

[114] P. Mattson *et al.*, "MLPerf Training Benchmark," *CoRR*, vol. abs/1910.01500, 2019. eprint: 1910.01500. [Online]. Available: http://arxiv.org/abs/1910.01500.

[115] P. Mattson, V. J. Reddi, C. Cheng, C. Coleman, G. Diamos, D. Kanter, P. Micikevicius, D. Patterson, G. Schmuelling, H. Tang, *et al.*, "MLPerf: An Industry Standard Benchmark Suite for Machine Learning Performance," *IEEE Micro*, vol. 40, no. 2, pp. 8–16, 2020.

[116] V. J. Reddi *et al.*, "MLPerf Inference Benchmark," in *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture*, ser. ISCA, 2020, pp. 446–459. DOI: 10.1109/ISCA45697.2020.00045.

[117] V. J. Reddi, C. Cheng, D. Kanter, P. Mattson, G. Schmuelling, and C.-J. Wu, "The Vision Behind MLPerf: Understanding AI Inference Performance," *IEEE Micro*, vol. 41, no. 3, pp. 10–18, 2021. DOI: 10.1109/MfM.2021.3066343.

[118] Z. Fan and E. Ma, "Predicting orientation-dependent plastic susceptibility from static structure in amorphous solids via deep learning," *Nature communications*, vol. 12, no. 1, pp. 1–13, 2021.

[119] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, *et al.*, "Highly accurate protein structure prediction with alphafold," *Nature*, vol. 596, no. 7873, pp. 583–589, 2021.

[120] J. Kates-Harbeck, A. Svyatkovskiy, and W. Tang, "Predicting disruptive instabilities in controlled fusion plasmas through deep learning," *Nature*, vol. 568, no. 7753, pp. 526–531, 2019.

[121] J. Thiyagalingam, M. Shankar, G. Fox, and T. Hey, "Scientific Machine Learning Benchmarks," *Nature Reviews Physics*, vol. 4, no. 6, pp. 413–420, 2022.

[122] J. Thiyagalingam, G. von Laszewski, J. Yin, M. Emani, J. Papay, G. Barrett, P. Luszczek, A. Tsaris, C. Kirkpatrick, F. Wang, T. Gibbs, V. Vishwanath, M. Shankar, G. Fox, and T. Hey, "Ai benchmarking for science: Efforts from the mlcommons science working group," in *High Performance Computing. ISC High Performance 2022 International Workshops*, H. Anzt, A. Bienz, P. Luszczek, and M. Baboulin, Eds., Cham: Springer International Publishing, 2022, pp. 47–64, ISBN: 978-3-031-23220-6.

[123] H. Wang, L. Zhang, J. Han, and W. E, "DeePMD-kit: A deep learning package for many-body potential energy representation and molecular dynamics," *Computer Physics Communications*, vol. 228, pp. 178–184, 2018, ISSN: 0010-4655. DOI: https://doi.org/10.1016/j.cpc.2018.03.016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0010465518300882.

[124] J. Zeng *et al.*, "DeePMD-kit v2: A software package for deep potential models," *The Journal of Chemical Physics*, vol. 159, no. 5, p. 054 801, Aug. 2023, ISSN: 0021-9606. DOI: 10.1063/5.0155600. eprint: https://pubs.aip.org/aip/jcp/article-pdf/doi/10.1063/5.0155600/18281511/054801_1_5.0155600.pdf. [Online]. Available: https://doi.org/10.1063/5.0155600.

[125] G. Derevyanko, G. Lamoureux, C. Outeiral, T. Oda, F. Fuchs, S. P. Mahajan, J. Moult, J. Haas, P. Maragakis, T. Ruzmetov, and M. AlQuraishi, *OpenFold2: Replicating AlphaFold2 in the Dark*, https://lupoglaz.github.io/OpenFold2/, 2023.

[126] B. Acun, A. Langer, E. Meneses, H. Menon, O. Sarood, E. Totoni, and L. V. Kalé, "Power, Reliability, and Performance: One System to Rule them All," *Computer*, vol. 49, no. 10, pp. 30–37, 2016. DOI: 10.1109/MC.2016.310.

[127] D. Chasapis, M. Casas, M. Moretó, M. Schulz, E. Ayguadé, J. Labarta, and M. Valero, "Runtime-Guided Mitigation of Manufacturing Variability in Power-Constrained Multi-Socket NUMA Nodes," in *Proceedings of the 2016 International Conference on Supercomputing*, ser. ICS '16, Istanbul, Turkey, 2016.

[128] D. Chasapis, M. Moretó, M. Schulz, B. Rountree, M. Valero, and M. Casas, "Power Efficient Job Scheduling by Predicting the Impact of Processor Manufacturing Variability," in *Proceedings of the ACM International Conference on Supercomputing*, ser. ICS '19, 2019, pp. 296–307, ISBN: 9781450360791. DOI: 10.1145/3330345.3330372. [Online]. Available: https://doi.org/10.1145/3330345.3330372.

[129] Y. Inadomi, T. Patki, K. Inoue, M. Aoyagi, B. Rountree, M. Schulz, D. Lowenthal, Y. Wada, K. Fukazawa, M. Ueda, M. Kondo, and I. Miyoshi, "Analyzing and Mitigating the Impact of Manufacturing Variability in Power-Constrained Supercomputing," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC, 2015, ISBN: 9781450337236. DOI: 10.1145/2807591.2807638. [Online]. Available: https://doi.org/10.1145/2807591.2807638.

[130] T. Patel, A. Wagenhäuser, C. Eibel, T. Hönig, T. Zeiser, and D. Tiwari, "What does Power Consumption Behavior of HPC Jobs Reveal? : Demystifying, Quantifying, and Predicting Power Consumption Characteristics," in *IEEE International Parallel and Distributed Processing Symposium*, ser. IPDPS, 2020, pp. 799–809. DOI: 10.1109/IPDPS47924.2020.00087.

[131] D. Skinner and W. Kramer, "Understanding the causes of performance variability in HPC workloads," in *Proceedings of the IEEE Workload Characterization Symposium*, ser. IISWC, 2005, pp. 137–149. DOI: 10.1109/IISWC.2005.1526010.

[132] T. Scogland, J. Azose, D. Rohr, S. Rivoire, N. Bates, and D. Hackenberg, "Node Variability in Large-Scale Power Measurements: Perspectives from the Green500, Top500 and EEHPCWG," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '15, Austin, Texas: Association for Computing Machinery, 2015, ISBN: 9781450337236. DOI: 10.1145/2807591.2807653. [Online]. Available: https://doi.org/10.1145/2807591.2807653.

[133] N. DeBardeleben, S. Blanchard, L. Monroe, P. Romero, D. Grunau, C. Idler, and C. Wright, "GPU Behavior on a Large HPC Cluster," in *Euro-Par 2013: Parallel Processing Workshops - BigDataCloud, DIHC, FedICI, HeteroPar, HiBB, LSDVE, MHPC, OMHI, PADABS, PROPER, Resilience, ROME, and UCHPC 2013, Aachen, Germany, August 26-27, 2013. Revised Selected Papers*, D. an Mey, M. Alexander, P. Bientinesi, M. Cannataro, C. Clauss, A. Costan, G. Kecskemeti, C. Morin, L. Ricci, J. Sahuquillo, M. Schulz, V. Scarano, S. L. Scott, and J. Weidendorfer, Eds., ser. Lecture Notes in Computer Science, vol. 8374, Springer, 2013, pp. 680–689. DOI: 10.1007/978-3-642-54420-0_66. [Online]. Available: https://doi.org/10.1007/978-3-642-54420-0_66.

[134] D. De Sensi, T. De Matteis, K. Taranov, S. Di Girolamo, T. Rahn, and T. Hoefler, "Noise in the Clouds: Influence of Network Performance Variability on Application Scalability," vol. 6, no. 3, Dec. 2022. DOI: 10.1145/3570609. [Online]. Available: https://doi.org/10.1145/3570609.

[135] F. Fraternali, A. Bartolini, C. Cavazzoni, and L. Benini, "Quantifying the Impact of Variability and Heterogeneity on the Energy Efficiency for a Next-Generation Ultra-Green Supercomputer," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 7, pp. 1575–1588, 2018. DOI: 10.1109/TPDS.2017.2766151.

[136] E. Sencan, D. Kulkarni, A. Coskun, and K. Konate, "Analyzing GPU Utilization in HPC Workloads: Insights from Large-Scale Systems," in *Practice and Experience in Advanced Research Computing 2025: The Power of Collaboration*, ser. PEARC, Association for Computing Machinery, 2025, ISBN: 9798400713989. DOI: 10.1145/3708035.3736010. [Online]. Available: https://doi.org/10.1145/3708035.3736010.

[137] P. Sinha, A. Guliani, R. Jain, B. Tran, M. D. Sinclair, and S. Venkataraman, "Not All GPUs Are Created Equal: Characterizing Variability in Large-Scale, Accelerator-Rich Systems," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, ser. SC, 2022, pp. 1–15.

[138] B. Topcu, D. Karabacak, and I. Oz, "Demystifying Power and Performance Variations in GPU Systems through Microarchitectural Analysis," *Computer Science and Information Systems*, vol. 22, no. 2, pp. 533–561, 2025. DOI: https://doi.org/10.2298/CSIS240722021T.

[139] X. You, Z. Xuan, H. Yang, Z. Luan, Y. Liu, and D. Qian, "GVARP: Detecting Performance Variance on Large-Scale Heterogeneous System," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis*, ser. SC, 2024.

[140] Z. Zhong, S. Sultanov, M. Papka, and Z. Lan, "Minimizing Power Waste in Heterogenous Computing via Adaptive Uncore Scaling," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, ser. SC, 2025.

[141] K. Heyman, *DRAM Thermal Issues Reach Crisis Point*, https://semiengineering.com/dram-thermal-issues-reach-crisis-point/, 2022.

[142] D. Stanzione, J. West, R. T. Evans, T. Minyard, O. Ghattas, and D. K. Panda, "Frontera: The evolution of leadership computing at the National Science Foundation," in *Practice and Experience in Advanced Research Computing*, 2020, pp. 106–111.

[143] TACC, *Texas Advanced Computing Center*, https://www.tacc.utexas.edu/, 2021.

[144] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *NIPS-W*, 2017.

[145] R. Jain, B. Tran, K. Chen, M. D. Sinclair, and S. Venkataraman, "PAL: A Variability-Aware Policy for Scheduling ML Workloads in GPU Clusters," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis*, ser. SC, Nov. 2024.

[146] J. Guerreiro, A. Ilic, N. Roma, and P. Tomás, "DVFS-aware application classification to improve GPGPUs energy efficiency," *Parallel Computing*, vol. 83, pp. 93–117, 2019, ISSN: 0167-8191. DOI: https://doi.org/10.1016/j.parco.2018.02.001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167819118300243.

[147] Science Feedback, *Training and Using ChatGPT Uses a Lot of Energy, but Exact Numbers Are Tricky to Pin Down Without Data from OpenAI*, 2024. [Online]. Available: https://science.feedback.org/training-and-using-chatgpt-uses-a-lot-of-energy-but-exact-numbers-are-tricky-to-pin-down-without-data-from-openai.

[148] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling Laws for Neural Language Models," *arXiv preprint arXiv:2001.08361*, 2020. [Online]. Available: https://arxiv.org/abs/2001.08361.

[149] U. E. I. Administration, *Average price of electricity to ultimate customers*, https://www.eia.gov/electricity/monthly/, 2025.

[150] U.S. Energy Information Administration, *Average annual electricity consumption for u.s. residential customers*, Available at https://www.eia.gov/, 2024.

[151] World Economic Forum, "Ai and energy: Will ai help reduce emissions or increase power demand? here's what to know," *NA*, Jul. 2024, (Accessed on 09/07/2025). [Online]. Available: https://www.weforum.org/stories/2024/07/generative-ai-energy-emissions/.

[152] D. Patterson, J. Gonzalez, Q. V. Le, C. Liang, L. Munguia, D. Rothchild, D. R. So, M. Texier, and J. Dean, "The Carbon Footprint of Large Neural Network Training," *arXiv preprint arXiv:2104.10350*, 2021. [Online]. Available: https://arxiv.org/abs/2104.10350.

[153] N. Jegham, M. Abdelatti, C. Y. Koh, L. Elmoubarki, and A. Hendawi, *How hungry is ai? benchmarking energy, water, and carbon footprint of llm inference*, 2025. arXiv: 2505.09598 [cs.CY]. [Online]. Available: https://arxiv.org/abs/2505.09598.

[154] Baeldung Editors, *How Much Energy Does ChatGPT Use?* 2023. [Online]. Available: https://www.baeldung.com/cs/chatgpt-large-language-models-power-consumption.

[155] T. B. Brown *et al.*, "Language Models are Few-Shot Learners," *arXiv preprint arXiv:2005.14165*, 2020. eprint: 2005.14165 (cs.CL). [Online]. Available: https://arxiv.org/abs/2005.14165.

[156] Data Science on Medium, *The Carbon Footprint of GPT-4*, 2023. [Online]. Available: https://medium.com/data-science/the-carbon-footprint-of-gpt-4-d6c676eb21ae.

[157] Extreme Networks, *Confronting AI's Growing Energy Appetite: Part 1*, 2023. [Online]. Available: https://www.extremenetworks.com/resources/blogs/confronting-ai-growing-energy-appetite-part-1.

[158] Epoch AI, *Why GPT-5 Used Less Training Compute Than GPT-4.5 (But GPT-6 Probably Won't)*, 2024. [Online]. Available: https://epoch.ai/gradient-updates/why-gpt5-used-less-training-compute-than-gpt45-but-gpt6-probably-wont.

[159] H. Editors, *AI's Dirty Secret: The Energy Cost of Training the Next GPT-5*, 2024. [Online]. Available: https://hackernoon.com/ais-dirty-secret-the-energy-cost-of-training-the-next-gpt-5.

[160] A. S. Bland, J. H. Rogers II, R. A. Kendall, D. B. Kothe, and G. M. Shipman, "Jaguar: The World's Most Powerful Computer," in *35th Cray User Group Meeting*, ser. CUG, May 2009.

[161] B. Bland, "Titan - Early experience with the Titan system at Oak Ridge National Laboratory," in *SC Companion: High Performance Computing, Networking Storage and Analysis*, 2012, pp. 2189–2211. DOI: 10.1109/SC.Companion.2012.356.

[162] D. E. Womble, M. Shankar, W. Joubert, J. T. Johnston, J. C. Wells, and J. A. Nichols, "Early Experiences on Summit: Data Analytics and AI Applications," *IBM Journal of Research and Development*, vol. 63, no. 6, 2:1–2:9, 2019. DOI: 10.1147/JRD.2019.2944146.

[163] S. Atchley *et al.*, "Frontier: Exploring Exascale The System Architecture of the First Exascale Supercomputer," in *International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC, 2023, pp. 1–16. DOI: 10.1145/3581784.3607089.

[164] T. R. W. Scogland, B. Subramaniam, and W. Feng, "Emerging Trends on the Evolving Green500: Year Three," in *Proc. IEEE IPDPS Workshops*, 2011, pp. 889–895. DOI: 10.1109/IPDPS.2011.229.

[165] A. Tschand, A. T. R. Rajan, S. Idgunji, A. Ghosh, J. Holleman, C. Kiraly, P. Ambalkar, R. Borkar, R. Chukka, T. Cockrell, O. Curtis, G. Fursin, M. Hodak, H. Kassa, A. Lokhmotov, D. Miskovic, Y. Pan, M. P. Manmathan, L. Raymond, T. S. John, A. Suresh, R. Taubitz, S. Zhan, S. Wasson, D. Kanter, and V. J. Reddi, "MLPerf Power: Benchmarking the Energy Efficiency of Machine Learning Systems from μWatts to MWatts for Sustainable AI," in *IEEE International Symposium on High Performance Computer Architecture*, ser. HPCA, 2025, pp. 1201–1216. DOI: 10.1109/HPCA61900.2025.00092.

[166] *Spec sert*^{2}, https://spec.org/sert2/, Server Efficiency Rating Tool with calibrated energy measurements, 2025. (visited on 05/06/2025).

[167] *Tpc-energy*, https://www.tpc.org/, Energy add-on kit for TPC database benchmarks, 2025. (visited on 05/06/2025).

[168] *Joulesort benchmark*, https://sortbenchmark.org/, Records sorted per Joule; storage I/O energy efficiency, 2025. (visited on 05/06/2025).

[169] *Hpcg-power*, https://hpcg-benchmark.org/, Energy efficiency (GFLOPS/W) for the High Performance Conjugate Gradient benchmark, 2025. (visited on 05/03/2025).

[170] *Hpl-mxp (hpl-ai)*, https://top500.org/news/hpl-ai-benchmark/, Mixed-precision LINPACK benchmark with GFLOPS/W metric, 2025. (visited on 05/03/2025).

[171] *Mlperf power: Training and inference*, https://mlcommons.org/en/power/, Joules, average Watts, Joules per sample/epoch for ML workloads, 2025. (visited on 05/06/2025).

[172] *Mlperf tiny: Energy mode*, https://mlcommons.org/en/tiny/, Microjoules per inference on microcontrollers, 2025. (visited on 05/06/2025).

[173] *Coremark-pro power*, https://www.eembc.org/coremarkpro/, Iterations per second per Watt for embedded/SoC devices, 2025. (visited on 05/06/2025).

[174] *Ul procyon ai inference power test*, https://benchmarks.ul.com/procyon, Images per Watt and fps/W on desktop and mobile devices, 2025. (visited on 05/06/2025).

[175] *Candle power study (sc19)*, https://doi.org/10.1145/3337821.3337924, Deep learning cancer benchmark with Joules/epoch & GFLOPS/W metrics, 2025. (visited on 05/03/2025).

[176] *Lulesh/minife energy benchmark*, https://doi.org/10.1109/ISPA-BDCloud-SocialCom-SustainCom55337.2022.00045, Energy/Joules per iteration for proxy-apps (Gerofi et al., 2022), 2025. (visited on 05/03/2025).

[177] *Exasmr power benchmark*, https://doi.org/10.1016/j.jpdc.2021.05.001, Energy vs accuracy trade-off for neutron transport mini-app, 2025. (visited on 05/03/2025).

[178] *Ee-hpc-wg energy benchmark (draft)*, https://eehpcwg.llnl.gov/, Community draft specification for node & job energy benchmarking, 2025. (visited on 05/03/2025).

[179] *Hpc-ai500 energy track (planned)*, https://www.hpc-ai.org/, Upcoming GFLOPS/W extension to HPC-AI500 mixed AI/HPC benchmark, 2025. (visited on 05/03/2025).

[180] *Parsec-3.1 energy extension*, https://parsec.cs.gatech.edu/, Research prototype adding power metrics to PARSEC benchmark suite, 2025. (visited on 05/03/2025).

[181] Prabhat *et al.*, "Scaling CosmoFlow to 15,000 GPUs and achieving 43 pflops," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC19)*, Includes CosmoFlow-Power joules/epoch data, 2019. DOI: 10.1145/3295500.3356175.

[182] K. Heitmann *et al.*, "The hacc framework: Energy and performance characterization," *Computing in Science & Engineering*, 2020, Adds HACC Energy Add-on joules/particle metric. DOI: 10.1109/MCSE.2020.3033659.

[183] T. Kurth *et al.*, "Exascale deep learning for climate analytics," in *International Conference for High Performance Computing (SC20)*, DeepCAM-Energy joules/epoch results, 2020. DOI: 10.5555/3433701.3433712.

[184] N. Wedi *et al.*, "Openifs energy benchmark report," ECMWF, Tech. Rep., 2023, kWh per model-day for full weather physics. [Online]. Available: https://www.ecmwf.int/en/publications/openifs/energy-benchmark.

[185] S. Páll *et al.*, "Gromacs-ee: Energy-efficient molecular dynamics on gpus," in *GPU Technology Conference (GTC)*, Introduces Joules/ns metric, 2024. [Online]. Available: https://developer.nvidia.com/gtc.

[186] A. Rodriguez *et al.*, "Energy delay product optimization of namd on summit," *Journal of Computational Chemistry*, 2019, Energy-Delay Product results for ApoA1. DOI: 10.1002/jcc.25785.

[187] P. Giannozzi *et al.*, "Energy-aware quantum espresso: Joules per scf step," in *International Workshop on Performance Modeling, Benchmarking and Simulation of High Performance Computing Systems (PMBS)*, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9955431.

[188] G. Kresse *et al.*, "Vasp power harness: Energy profiling of dft md," Vienna University of Technology, Tech. Rep., 2023. [Online]. Available: https://vasp.at/energy-harness.

[189] R. Jain *et al.*, "Characterizing energy consumption of openfoam on modern hpc systems," in *Workshop on Energy Efficient Supercomputing*, 2021. DOI: 10.1145/3489059.3494181.

[190] T. Farr *et al.*, "Insar-ai: Power characterization of satellite image unwrapping," *IEEE Journal of Selected Topics in Applied Earth Observations*, 2024, Joules per satellite scene. DOI: 10.1109/JSTARS.2024.1234567.

[191] G. Fox *et al.*, "H3d: Hydrology 3d energy benchmark," in *International Conference on Computational Science (ICCS)*, Joules per timestep metric, 2023. [Online]. Available: https://iccs2023.org.

[192] *Spec ptdaemon / sert energy for hpc*, https://spec.org/ptdaemon/, Calibrated power logging used with SPEC benchmarks on HPC systems, 2025. (visited on 05/03/2025).

[193] *Scaphandre*, https://github.com/hubblo-org/scaphandre, Process & node power telemetry agent for Linux clusters (Watts, kWh), 2025. (visited on 05/03/2025).

[194] *Kepler: Kubernetes-based energy profiler*, https://github.com/sustainable-computing-io/kepler, Watts and Joules per container/pod using eBPF/RAPL, 2025. (visited on 05/06/2025).

[195] *Codecarbon*, https://codecarbon.io/, Process-level kWh and kg CO2e estimation library, 2025. (visited on 05/06/2025).

[196] *Carbontracker*, https://github.com/lfwa/carbontracker, Energy and CO2 prediction for deep-learning training, 2025. (visited on 05/06/2025).

[197] *Powerpack / mont-blanc*, https://gitlab.bsc.es/mont-blanc/PowerPACK, Energy & power profiling toolkit for MPI/OpenMP mini-apps (Joules, Watts), 2025. (visited on 05/03/2025).

[198] *Cray pat energy counters*, https://support.hpe.com/hpesc/public/docDisplay?docId=a00111513en_us, Integrated energy-per-function profiling in HPE/Cray Performance Analysis Tool, 2025. (visited on 05/03/2025).

[199] *Ibm powerapi (pmlib)*, https://github.com/IBM/powerapi, System & per-process kWh reporting on Power-based supercomputers, 2025. (visited on 05/03/2025).

[200] *Nvidia dcgm energy*, https://developer.nvidia.com/dcgm, GPU Joules & Watts via Data Center GPU Manager; attachable to HPC benchmarks, 2025. (visited on 05/03/2025).

[201] *Intel vtune power analysis*, https://www.intel.com/content/www/us/en/developer/tools/oneapi/vtune-profiler.html, Package Watts & energy per function for MPI/OpenMP codes, 2025. (visited on 05/03/2025).

[202] G. von Laszewski, *Cloudmesh gpu monitor*, [Online; accessed 2025-11-26], Feb. 2022. [Online]. Available: https://github.com/cloudmesh/cloudmesh-gpu.

[203] M. Khairy, Z. Shen, T. M. Aamodt, and T. G. Rogers, "Accel-Sim: An Extensible Simulation Framework for Validated GPU Modeling," in *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture*, ser. ISCA, 2020, pp. 473–486. DOI: 10.1109/ISCA45697.2020.00047.

[204] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, "The gem5 simulator," *ACM SIGARCH Computer Architecture News*, vol. 39, no. 2, pp. 1–7, 2011.

[205] J. Lowe-Power *et al.*, "The gem5 simulator: Version 20.0+," *CoRR*, vol. abs/2007.03152, 2020. eprint: 2007.03152 (cs.AR).

[206] A. F. Rodrigues, K. S. Hemmert, B. W. Barrett, C. Kersey, R. Oldfield, M. Weston, R. Risen, J. Cook, P. Rosenfeld, E. Cooper-Balis, and B. Jacob, "The Structural Simulation Toolkit," *SIGMETRICS Perform. Eval. Rev.*, vol. 38, no. 4, pp. 37–42, Mar. 2011, ISSN: 0163-5999. DOI: 10.1145/1964218.1964225. [Online]. Available: https://doi.org/10.1145/1964218.1964225.

[207] S. Nema, R. Razdan, A. Rodrigues, K. Hemmert, G. Voskuilen, D. Adak, S. Hammond, A. Awad, and C. Hughes, "ERAS: Enabling the Integration of Real-World Intellectual Properties (IPs) in Architectural Simulators," *Sandia National Labs Tech Report*, Sep. 2021. DOI: 10.2172/1854734. [Online]. Available: https://www.osti.gov/biblio/1854734.

[208] W. Brewer, M. Maiterth, V. Kumar, R. Wojda, S. Bouknight, J. Hines, W. Shin, S. Greenwood, D. Grant, W. Williams, and F. Wang, "A digital twin framework for liquid-cooled supercomputers as demonstrated at exascale," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, 2024.

[209] A. Bakhoda, G. L. Yuan, W. W. L. Fung, H. Wong, and T. M. Aamodt, "Analyzing CUDA workloads using a detailed GPU simulator," in *2009 IEEE International Symposium on Performance Analysis of Systems and Software*, ser. ISPASS, Apr. 2009, pp. 163–174. DOI: 10.1109/ISPASS.2009.4919648.

[210] V. Kandiah, S. Peverelle, M. Khairy, A. Manjunath, J. Pan, T. G. Rogers, T. M. Aamodt, and N. Hardavellas, "AccelWattch: A Power Modeling Framework for Modern GPUs," in *Proceedings of the 54th IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO, Oct. 2021.

[211] J. Lew, D. Shah, S. Pati, S. Cattell, M. Zhang, A. Sandhupatla, C. Ng, N. Goli, M. D. Sinclair, T. G. Rogers, and T. M. Aamodt, "Analyzing Machine Learning Workloads Using a Detailed GPU Simulator," *CoRR*, vol. abs/1811.08933, 2018. eprint: 1811.08933. [Online]. Available: http://arxiv.org/abs/1811.08933.

[212] C. Avalos Baddouh, M. Khairy, R. N. Green, M. Payer, and T. G. Rogers, "Principal Kernel Analysis: A Tractable Methodology to Simulate Scaled GPU Workloads," in *54th Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO '21, Virtual Event, Greece: Association for Computing Machinery, 2021, pp. 724–737, ISBN: 9781450385572. DOI: 10.1145/3466752.3480100. [Online]. Available: https://doi.org/10.1145/3466752.3480100.

[213] A. Gutierrez, B. M. Beckmann, A. Dutu, J. Gross, M. LeBeane, J. Kalamatianos, O. Kayiran, M. Poremba, B. Potter, S. Puthoor, M. D. Sinclair, M. Wyse, J. Yin, X. Zhang, A. Jain, and T. Rogers, "Lost in Abstraction: Pitfalls of Analyzing GPUs at the Intermediate Language Level," in *24th IEEE International Symposium on High Performance Computer Architecture*, ser. HPCA, Feb. 2018, pp. 608–619. DOI: 10.1109/HPCA.2018.00058.

[214] S. Rogers, J. Slycord, M. Baharani, and H. Tabkhi, "gem5-SALAM: A System Architecture for LLVM-based Accelerator Modeling," in *53rd Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO, 2020, pp. 471–482. DOI: 10.1109/MICRO50266.2020.00047.

[215] Z. Spencer, S. Rogers, J. Slycord, and H. Tabkhi, "Expanding Hardware Accelerator System Design Space Exploration with gem5-SALAMv2," *Journal of Systems Architecture*, vol. 154, p. 103 211, 2024, ISSN: 1383-7621. DOI: https://doi.org/10.1016/j.sysarc.2024.103211. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1383762124001486.

[216] A. Chaudhari and M. D. Sinclair, "Toward Full-System Heterogeneous Simulation: Merging gem5-SALAM with Mainline gem5," in *6th gem5 Users' Workshop*, Jun. 2025.

[217] A. Smith, B. Bruce, J. Lowe-Power, and M. D. Sinclair, "Designing Generalizable Power Models For Open-Source Architecture Simulators," in *3rd Open-Source Computer Architecture Research Workshop*, ser. OSCAR, 2024.

[218] V. Ramadas, M. Poremba, B. M. Beckmann, and M. D. Sinclair, "Improving gem5's GPU FS Support," in *The 5th gem5 Users' Workshop*, Jun. 2023.

[219] V. Ramadas, M. Poremba, B. M. Beckmann, and M. D. Sinclair, "Simulation Support for Fast and Accurate Large-Scale GPGPU and Accelerator Workloads," in *3rd Open-Source Computer Architecture Research Workshop*, ser. OSCAR, 2024.

[220] V. Ramadas and M. D. Sinclair, "Simulating Machine Learning Models at Scale," in *SRC TECHCON*, Sep. 2024.

[221] C. Hughes, S. D. Hammond, R. J. Hoekstra, M. Zhang, Y. Liu, and T. Rogers, "SST-GPU: A Scalable SST GPU Component for Performance Modeling and Profiling," *Sandia National Lab*, Jan. 2021. DOI: 10.2172/1762830. [Online]. Available: https://www.osti.gov/biblio/1762830.

[222] C. Hughes, S. D. Hammond, M. Khairy, M. Zhang, R. Green, T. Rogers, and R. J. Hoekstra, "Balar: A SST GPU Component for Performance Modeling and Profiling," *Sandia National Lab*, Sep. 2019. DOI: 10.2172/1560919. [Online]. Available: https://www.osti.gov/biblio/1560919.

[223] M. Hsieh, K. Pedretti, J. Meng, A. Coskun, M. Levenhagen, and A. Rodrigues, "SST + Gem5 = a Scalable Simulation Infrastructure for High Performance Computing," in *Proceedings of the 5th International ICST Conference on Simulation Tools and Techniques*, ser. SIMUTOOLS '12, Desenzano del Garda, Italy: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2012, pp. 196–201, ISBN: 9781450315104.

[224] H. Nguyen and J. Lowe-Power, "gem5/SST Integration 2021: Scaling Full-system Simulations," in *The 4th gem5 Users' Workshop with ISCA*, 2022.

[225] M. Maiterth, W. H. Brewer, J. S. Kuruvella, A. Dey, T. Z. Islam, K. Menear, D. Duplyakin, R. Kabir, T. Patki, T. Jones, *et al.*, "HPC digital twins for evaluating scheduling policies, incentive structures and their impact on power and cooling," in *SC25-W: Workshops of the International Conference for High Performance Computing, Networking, Storage and Analysis*, IEEE, 2025.

[226] W. Brewer, M. Maiterth, and D. Fay, "Trace replay simulation of MIT SuperCloud for studying optimal sustainability policies," in *2025 IEEE High Performance Extreme Computing Conference (HPEC)*, IEEE, 2025.

[227] S. Kalepu, W. H. Brewer, M. Maiterth, and R. Vuduc, "Virtual benchmarking for HPC systems using ExaDigiT and Calculon," in *2025 IEEE High Performance Extreme Computing Conference (HPEC)*, IEEE, 2025.

[228] G. von Laszewski, B. Hawks, M. Colombo, R. Shiraishi, A. Krishnan, N. Tran, and G. C. Fox, *Mlcommons science working group ai benchmarks collection*, GitHub, Online Collection: =https://mlcommons-science.github.io/benchmark/, Jun. 2025. [Online]. Available: https://mlcommons-science.github.io/benchmark/benchmarks.pdf.

Appendix A. Abbreviations

A, P Application and Parameters – Formal components defining a scientific task $T = (A, P)$.

AI Artificial Intelligence – Computational systems performing tasks that typically require human intelligence, such as learning, reasoning, or perception.

API Application Programming Interface – A set of functions and protocols for building and integrating software applications.

ARC AI2 Reasoning Challenge – Benchmark for commonsense and science question answering at the K–12 level.

AristoBench A benchmark based on U.S. science exam questions.

B Benchmark – A formal specification of an evaluation setup $B = (I, D, T, M, C_B, R)$.

BioASQ A benchmark for biomedical question answering and document retrieval.

BSP Bulk synchronous programming

CAPEX/OPEX Capital Expenditure / Operating Expenditure – Financial measures of investment and operational costs.

CarbonTracker Tool for tracking and reporting ML training energy use and CO_2 emissions.

C_B, C_c Benchmark and Component Constraints – Rules applied to benchmarks or components for fairness, reproducibility, and comparability.

CodeCarbon Python package estimating CO_2 emissions based on compute energy and grid intensity.

CO_2e Carbon Dioxide Equivalent – Standardized unit comparing greenhouse gas emissions by global warming potential.

cPUE Compute-Only Power Usage Effectiveness – Refinement of PUE focusing on compute power versus total facility power.

CORE-Bench A benchmark for computational reproducibility.

CPU Central Processing Unit – The main processor of a computer, optimized for sequential execution.

CSR Corporate Social Responsibility – A framework for a company's environmental and social accountability.

Data Carpentry A Carpentries initiative focused on teaching data management and analysis skills.

DCGM Data-Centre GPU Manager – NVIDIA's toolkit for GPU monitoring, management, and power measurement.

DCiE Data-Centre-infrastructure Efficiency – Metric defined as $P_{\mathrm{IT}}/P_{\mathrm{fac}}$, the reciprocal of PUE.

DeepONet Deep Operator Network – A neural architecture for learning operators mapping between function spaces.

DOE U.S. Department of Energy – Federal agency supporting large-scale AI and HPC research.

DOI Digital Object Identifier – Persistent identifier for published datasets, software, and benchmarks.

DVFS Dynamic Voltage and Frequency Scaling – Power management technique adjusting CPU/GPU frequency to balance performance and energy.

EDP Energy–Delay Product – A metric combining performance (time-to-solution) and energy efficiency.

EIA Energy Information Administration – U.S. agency providing official energy statistics and analysis.

EE-HPC WG Energy Efficient HPC Working Group – Community standardizing energy measurement and reporting in HPC.

ElectricityMap API providing real-time carbon intensity of electricity grids worldwide.

Energy-to-Solution Total energy consumed to complete

a computational task.

EU European Union – Political and economic union of 27 member states.

FAIR Findable, Accessible, Interoperable, Re-usable – Principles for data and software management.

FNO Fourier Neural Operator – Deep learning architecture for solving partial differential equations efficiently.

Galactica Eval A benchmark measuring scientific writing and knowledge recall capabilities.

GFLOPS Giga Floating-Point Operations per Second – Measure of computational performance.

GFLOPS/W GFLOPS per Watt – Energy efficiency metric used in Green500.

GPU Graphics Processing Unit – Specialized processor for graphics and parallel computation, widely used in ML and HPC.

Green500 Ranking of the most energy-efficient supercomputers in the world.

HPC High-Performance Computing – The use of supercomputers and parallel processing to solve large, complex problems.

HPC AI500 Benchmark suite evaluating deep learning performance at HPC scale.

HPC Carpentry Carpentry modules teaching HPC skills and workflows.

HPCG High Performance Conjugate Gradients — a benchmark that represents real-world scientific workloads, such as partial differential equations' solvers, better than the LINPACK benchmark.

HPCG-Power Energy-aware version of the HPCG benchmark.

HPL High-Performance LINPACK Benchmark – Measures floating-point computing power.

HPL-MxP Mixed-Precision variant of the LINPACK benchmark that uses multiple floating-point formats for improved FLOP and energy metrics.

IO500 Benchmark suite evaluating HPC storage performance.

IOPS Input/Output Operations Per Second – Metric measuring storage system performance.

I Infrastructure – Hardware and software environment required to execute a benchmark.

J/epoch Joules per training epoch – ML training energy efficiency metric.

J/sample Joules per sample – Energy consumed per inference or training sample.

J/step Joules per simulation step – Energy metric for iterative simulation workloads.

JouleSort Benchmark measuring energy efficiency of sorting tasks.

Kaggle Online platform hosting AI and data science competitions.

Kepler Kubernetes-based Efficient Power Level Exporter – Collects power metrics in containerized workloads.

KPI Key Performance Indicator – Quantitative metric for performance or efficiency.

kg CO_2e Kilograms of CO_2 equivalent – Emission unit measuring carbon footprint.

kg CO_2e/job Kilograms CO_2e emitted per completed computational job.

kWh Kilowatt-hour – Standard unit of electrical energy.

LAB-Bench Benchmark evaluating AI models on laboratory research assistance tasks.

LLM Large Language Model – AI model trained on massive text corpora, capable of natural language generation.

ML Machine Learning – A subset of AI focused on algorithms that improve automatically through data.

MLCommons Open engineering consortium developing ML benchmarks, datasets, and best practices.

MLflow Open-source platform for managing the ML lifecycle, including experimentation and deployment.

MLPerf MLCommons benchmark suite for ML training and inference performance.

MLPerf HPC MLPerf extension for measuring ML workloads in HPC environments.

MLPerf Power MLPerf extension tracking energy efficiency (e.g., Joules/sample).

MLPerf Tiny Power MLPerf subset focused on energy use in constrained edge devices.

MMLU Massive Multitask Language Understanding benchmark – Evaluates cross-domain academic knowledge.

MPI Message Passing Interface – Communication standard for parallel programming in distributed-memory systems.

M Metrics – Quantitative measures such as accuracy, throughput, latency, or energy.

NVML NVIDIA Management Library – API for monitoring and managing NVIDIA GPU performance and power.

OmniStat AMD's monitoring and management interface, similar to NVIDIA's NVML.

P_{fac} Facility Power

P_{IT} IT Equipment Power

PDEBench Benchmark for solving partial differential equations using ML and operator learning.

PM_COUNTER Node-level power counter (e.g., HPE-Cray) for system-level power tracking.

PowerAPI Software library and framework for energy measurement and analysis.

PPA Power-Purchase Agreement – Contract defining terms for renewable energy procurement.

PUE Power Usage Effectiveness – Data center efficiency metric $P_{\text{fac}}/P_{\text{IT}}$.

PTDaemon SPEC Power Temperature Daemon – Tool for calibrated power measurement in benchmarks.

PyTorch Profiler Tool for performance analysis of PyTorch models.

QA Question Answering – NLP task where systems respond to questions posed in natural language.

R Programming language for statistical computing and

data analysis.

RAPL Running Average Power Limit – Intel/AMD interface for on-chip power measurement.

RFP Request for Proposal – Formal request soliciting project bids or benchmark implementations.

R Results – Outputs of a benchmark including accuracy, energy, and performance metrics.

SCIQ Science Question Answering benchmark focusing on inference and reasoning.

SciEval Benchmark for multi-domain scientific reasoning.

SciSafeEval Benchmark measuring safety alignment of AI systems in scientific contexts.

Scaphandre Open-source energy monitoring agent integrating with Prometheus.

SERT Server Efficiency Rating Tool – SPEC benchmark for server energy efficiency.

Slurm Simple Linux Utility for Resource Management – Widely used HPC workload manager.

SM Streaming Multiprocessor – GPU core building block for parallel computation.

SME Small and Medium-sized Enterprise – Business classification by size and revenue.

Software Carpentry Community teaching foundational computing skills for researchers.

SKU Stock-Keeping Unit

SPEC Standard Performance Evaluation Corporation – Organization developing performance benchmarks.

SPEC HPC SPEC benchmark suite for HPC system performance.

SPECpower SPEC benchmark suite measuring server energy efficiency.

SQL Structured Query Language – Standard for managing and querying relational databases.

STREAM Synthetic benchmark measuring sustainable memory bandwidth.

T Scientific Task – Core task evaluated by a benchmark, e.g., classification or prediction.

TDP Thermal Design Power – Maximum heat generated by a processor under typical workloads.

TensorBoard Visualization toolkit for TensorFlow training and profiling.

TorchInfo Python package summarizing PyTorch models (layers, parameters, memory use).

TOP500 Official list of the 500 most powerful supercomputers in the world.

TPC-Energy Benchmark suite for transaction workloads with energy measurement.

TVA Tennessee Valley Authority – U.S. regional energy provider supporting grid research.

Vampir Performance analysis tool for parallel applications with detailed execution tracing.

VTune Intel VTune Profiler – Performance and energy analysis tool for fine-grained profiling.

WattTime API providing marginal emissions signals for carbon-aware workload scheduling.

Wh/DB-phase Watt-hours per database phase – Metric used in TPC-Energy benchmarks.

Appendix B. Contributions

Gregor von Laszewski is the lead author of the paper. He identified first that efforts in benchmark carpentry and democratization are needed. He has lead the organization of this paper in the MLCommons Science Working Group. He also created the initial version of [228] which is related and relevant to this effort.

Piotr Luszczek has contributed to integration of many decades of experiences from designing, implementing, running, and collecting results from HPC benchmarks.

Wesley Brewer has contributed to the simulation section.

Jeyan Thiyagalingam has worked on the GPU benchmarking section. Reviewed the paper and made corrections.

Juri Papay has worked on the GPU benchmarking section. Updated the GPU HW details of MLCommon benchmarks.

Geoffrey C. Fox is leading the MLCommons Science Working group and has contributed to many of the ideas. The experiences and discussions with Gregor von Laszewski around improvements to the earthquake benchmark have significantly contributed to this effort. The educational effort of using the earthquake benchmark with a number of students motivated this effort.

Armstrong Foundjem has provided an early version and led the Energy section, and has contributed to the paper writing and overall improvement.

Gregg Barrett has participated in discussions as part of the working group meetings and contributed to an early version of this paper.

Murali Emani has participated in discussions as part of the working group meetings and improved the article.

Shirley V. Moore has written text for the Profiling and Performance Analysis Section.

Vijay Janapa Reddi has participated in discussions as part of the working group meetings and improved the article.

Matthew D. Sinclair, Shivaram Venkataraman and Rutwik Jain participated in discussions as part of the working group meetings and wrote the variability section of the article. Sinclair also wrote the simulation section of the paper and helped improve the paper in other sections.

Christine Kirkpatrick has worked on conceptualizing the ideas and discussion, and helping with the Carpentries background section.

Kartik Mathur Has worked on improving an early version of the Energy section.

Victor Lu Has participated in writing the paper.

Tianhao Li has participated in discussions as part of the working group meetings and participated in identification of limitations of current benchmarks.

Sebastian Lobentanzer Has participated in the discussions in the working group and has contributed to the abstract, intro, definitions, formalization, and benchmark sections with content and editing.

Sujata Goswami has worked on the MLCommons benchmark details in Table I.
Abdulkareem Alsudais has reviewed the motivation to AI Benchmark Carpentry and contributed to the writing of this paper.
Kongtao Chen has worked on the monitoring sections, related benchmarks, and participated in discussions as part of the working group meetings.
Tejinder Singh has edited and improved AI hardware benchmarking and infrastructure sections and provided new KPIs for AI hardware benchmarking.
Kirsten Morehouse knmorehouse@gmail.com has participated in discussions as part of the working group meetings. Morehouse also reviewed the paper and made improvements.
Marco Colombo, Benjamin Hawks, and Nhan Tran have worked on the benchmark ontology and Table III.
Khojasteh Z. Mirza has participated in discussions as part of the working group meetings and worked an a very early version of the energy section.
Renato Umeton revised the manuscript for consistency and coherence.
Sasidhar Kunapuli and Gavin Farrell gavinmichael.farrell@phd.unipd.it have participated in discussions as part of the working group meetings.