[BenchCouncil Transactions on Benchmarks, Standards and Evaluations 2 (2022) 100083](https://doi.org/10.1016/j.tbench.2022.100083)



Contents lists available at [ScienceDirect](https://www.keaipublishing.com/en/journals/benchcouncil-transactions-on-benchmarks-standards-and-evaluations/)

BenchCouncil Transactions on Benchmarks,

Standards and Evaluations

journal homepage: [www.keaipublishing.com/en/journals/benchcouncil-transactions-on-](http://www.keaipublishing.com/en/journals/benchcouncil-transactions-on-benchmarks-standards-and-evaluations/) [benchmarks-standards-and-evaluations/](http://www.keaipublishing.com/en/journals/benchcouncil-transactions-on-benchmarks-standards-and-evaluations/)

Research Article

[](http://crossmark.crossref.org/dialog/?doi=10.1016/j.tbench.2022.100083&domain=pdf)HPC AI500 V3.0: A scalable HPC AI benchmarking framework

Zihan Jiang [a](#_bookmark0),[b](#_bookmark1),[∗](#_bookmark2), Chunjie Luo [a](#_bookmark0), Wanling Gao [a](#_bookmark0), Lei Wang [a](#_bookmark0), Jianfeng Zhan [a](#_bookmark0),[b](#_bookmark1)

a *Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China*

b *University of Chinese Academy of Sciences, Beijing, China*

A R T I C L E I N F O A B S T R A C T

*Keywords:*

Artificial intelligence

High performance computing Benchmarking

Scalability

In recent years, the convergence of High Performance Computing (HPC) and artificial intelligence (AI) makes the community desperately need a benchmark to guide the design of next-generation scalable HPC AI systems. The success of the HPL benchmarks and the affiliated TOP500 ranking indicates that scalability is the fundamental requirement to evaluate HPC systems. However, being scalable in terms of these emerging AI workloads like deep learning (DL) raises nontrivial challenges. This paper formally and systematically analyzes the factor that limits scalability in DL workloads and presents HPC AI500 v3.0, a scalable HPC AI benchmarking framework. The HPC AI500 V3.0 methodology is inspired by bagging, which utilizes the collective wisdom of an ensemble of base models and enables the benchmarks to be adaptively scalable to different scales of HPC systems. We implement HPC AI500 V3.0 in a highly customizable manner, maintaining the space of various optimization from both system and algorithm levels. By reusing the representative workloads in HPC AI500 V2.0, we evaluate HPC AI500 V3.0 on typical HPC systems, and the results show it has near-linear scalability. Furthermore, based on the customizable design, we present a case study to perform a trade-off between AI model quality and its training speed. The source code of HPC AI500 V3.0 is publicly available from the HPC AI500 project homepage <https://www.benchcouncil.org/aibench/hpcai500/>.

# Introduction

*D*eep *L*earning (DL) has been a dominating technology in *A*rtificial *I* ntelligence (AI) as its huge success in many challenging AI problems, such as image classification [[1](#_bookmark27)–[3](#_bookmark29)], object detection [[4](#_bookmark30)–[6](#_bookmark32)], and natural language processing [[7](#_bookmark33)–[9](#_bookmark34)]. DL allows building a computational model composed of multiple processing layers with trainable weights to learn the presentation of data [[10](#_bookmark35)]. To harness larger datasets and achieve higher model quality (e.g., Top1 accuracy), in recent years, tremendous DL models have been proposed endlessly, both for commercial applica- tions [[11](#_bookmark36)–[16](#_bookmark38)] and scientific computing [[17](#_bookmark39)–[20](#_bookmark42)]. These giant models usually have deeper layers and billions of weights, which is extremely computation-intensive. Hence, academia and industry are greatly in- terested in designing and building next-generation HPC systems to run these emerging AI workloads for their computation requirement [[21](#_bookmark43), [22](#_bookmark44)]. Benchmark plays an important role in this process, as it provides the input and methodology for evaluation [[23](#_bookmark45)].

In the past three decades, the HPL benchmark [[24](#_bookmark46)] and the affiliated TOP500 ranking [[25](#_bookmark47)] witnessed the thriving of HPC systems. From CM-5 (1993) [[26](#_bookmark48)] to Fugaku (2020) [[27](#_bookmark49)], the FLOPS performance of the NO.1 supercomputer on the TOP500 list improves by more than

106×. HPL has become the measurement standard [[28](#_bookmark50)] in the HPC field

for thirty years and will continue to be. The reason for its success is

twofold. On the one hand, HPL solves a (random) dense linear system in double precision, which captures the general characteristic that many scientific applications share. We conclude this property as *relevancy*. On the other hand, HPL can adapt to scalable systems by adjusting the input matrix size. We summarize this property as *scalability*. The HPL lesson indicates that *relevancy* and *scalability* are two significant properties for an ideal benchmark. Most of the previous work [[29](#_bookmark51)–[34](#_bookmark53)] in AI benchmarking focus on relevancy and select represent workloads in real-world AI applications. However, they ignored the scalability issue.

Scalability is difficult to guarantee for AI workloads. According to the experiences in the previous researches [[36](#_bookmark55),[46](#_bookmark60)], each AI workload has the best training batchsize, which is irrelevant to the system scale, to achieve state-of-the-art quality. This observation indicates that no matter how the scale of the system changes, the amount of parallel computation processed remains the same. Although many system op- timizations [[13](#_bookmark37),[47](#_bookmark61)–[52](#_bookmark63)] are proposed, all they can do is process this constant amount of computation as fast as possible by utilizing various parallel techniques (e.g., data parallelism [[53](#_bookmark64)]). Therefore, with the continuous growth of system scale, the speed of training existing AI workloads is rapidly accelerated. As shown in [Fig.](#_bookmark4) [1](#_bookmark4), from 2017 to 2021, with the development of HPC AI systems, the training time of

∗ Corresponding author at: Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China.

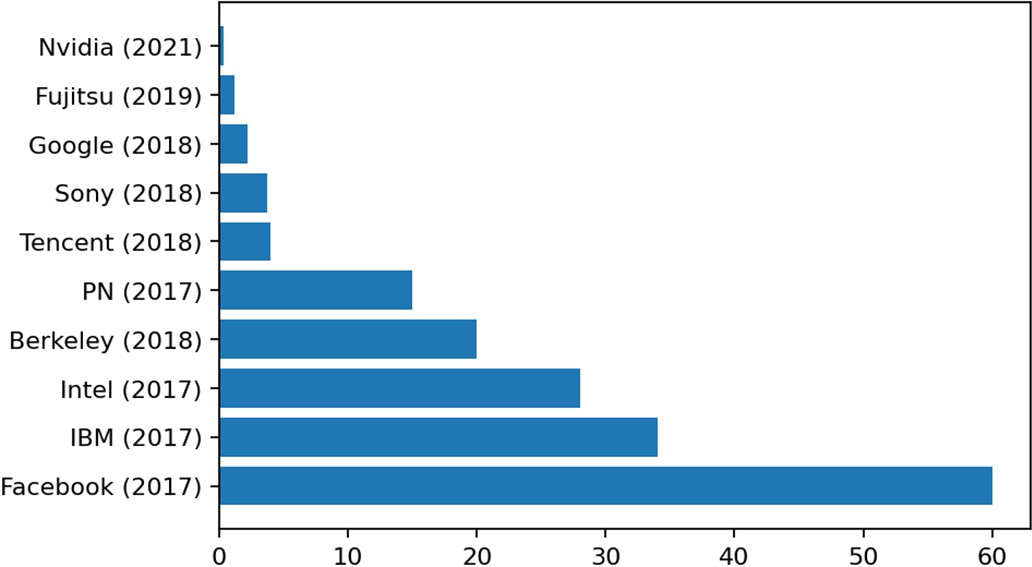
*E-mail addresses:* [jiangzihan@ict.ac.cn](mailto:jiangzihan@ict.ac.cn) (Z. Jiang), [luochunjie@ict.ac.cn](mailto:luochunjie@ict.ac.cn) (C. Luo), [gaowanling@ict.ac.cn](mailto:gaowanling@ict.ac.cn) (W. Gao), [wanglei\_2011@ict.ac.cn](mailto:wanglei_2011@ict.ac.cn) (L. Wang), [zhanjianfeng@ict.ac.cn](mailto:zhanjianfeng@ict.ac.cn) (J. Zhan).

<https://doi.org/10.1016/j.tbench.2022.100083>

Received 17 November 2022; Received in revised form 23 December 2022; Accepted 23 December 2022

Available online 29 December 2022

2772-4859/© 2022 The Authors. Publishing services by Elsevier B.V. on behalf of KeAi Communications Co. Ltd. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

* We evaluate HPC AI500 V3.0 by reusing HPC AI500 v2.0 work- loads on typical HPC systems to show its scalability and customiz- ability (Section [4](#_bookmark20)).

# Background and challenge

* 1. *Deep learning preliminary*

The whole training process of modern DL models is essentially a non-convex optimization. Mathematically, it can be represented as:

1 ∑*𝑁*

min *𝑓* (*𝑥*) ∶=

∀*𝑥*∈R*𝑛*

*𝑁 𝑖*=1

*𝑓𝑖*(*𝑥*)*,* (1)

academia [[35](#_bookmark54)] and industry [[36](#_bookmark55)–[44](#_bookmark58)]. PN refers to Preferred Networks [[45](#_bookmark59)]. The *𝑥*-axis **/ig. 1.** ImageNet/ResNet-50 is a popular showcase for optimizing HPC AI systems from

refers to the training time measured in minutes.

ResNet-50 [[2](#_bookmark28)] has dropped exponentially, and the result of Nvidia [[44](#_bookmark58)] shows that it now can be done in under half a minute. From the benchmarking perspective, such a short running time does not allow for a thorough and endurable evaluation. Furthermore, the fixed amount of computation is distributed on the HPC system with a growing scale, which makes the resource utilization of each computing node extremely unsaturated.

Two prior works attempt to address the scalability problem in HPC AI benchmarking, namely AIPerf [[54](#_bookmark65)] and HPL-AI [[55](#_bookmark66)]. However, they

both have their own flaws. AIPerf uses network architecture search

where *𝑓𝑖* is a loss function for data point *𝑖* ∈ {1*,* 2*,* 3*,* … *, 𝑁* }, which measures the deviation of the model prediction from the data. *𝑥* is the

vector of weights being optimized. The process of optimizing the loss function is called training and is performed iteratively.

* + 1. *Mini-batch stochastic gradient descent*

ing DL models. Vanilla SGD updates weight *𝑥* by adding the gradient Stochastic Gradient Descent (SGD) is the dominant method for train-

computed on a single data point of the whole dataset. Since only one random data point is processed at one iteration, this approach has two disadvantages. First, such a noisy update makes the training process unstable [[62](#_bookmark73)]. Second, the computation is inefficient, especially when using computing devices such as GPUs. Mini-batch SGD is proposed

to remedy these two deficiencies. It minimizes the loss function *𝑓*

iteratively in the following form:

(NAS) [[56](#_bookmark67)] as the primary workload. NAS automatically searches the network architecture with a predefined probability, introducing ran-

*𝑥𝑘*+1 = *𝑥𝑘* − *𝜂𝑘*

( 1 ∑

|*𝐵* |

)

∇*𝑓𝑖*(*𝑥𝑘*)

(2)

domness to the benchmarking process. HPL-AI allows mixed-precision LU decomposition to solve a linear equation system and tends to be irrelevant to most AI workloads [[57](#_bookmark68)].

Bagging (Bootstrap Aggregation) [[58](#_bookmark69)] is designed to improve the stability and quality of the prediction by utilizing the collective wisdom of an ensemble of base models. As a meta-algorithm of ensemble learn- ing [[59](#_bookmark70)], a critical feature of bagging is the independence between each base model. This independence makes bagging can be implemented as a highly parallel way to scale out with the number of nodes in an HPC system. Another merit of bagging is its flexibility and not being bound to any AI algorithm. In other words, we can easily and quickly achieve relevancy by integrating a state-of-the-art or state-of-the-practice algo- rithm into our bagging-based benchmarking framework. Considering the advantages above, this paper presents a bagging-based scalable AI benchmarking framework, which we call HPC AI500. HPC AI500 V3.0 extends our previous works: HPC AI500 V1.0 [[50](#_bookmark62)] and HPC AI500 V2.0 [[57](#_bookmark68)]. [Table](#_bookmark6) [1](#_bookmark6) summarizes the differences between HPC AI V3.0 from the other related works. HPC AI500 V3.0 not only leverages the advantages of bagging to achieve scalability and relevancy but also maintains user-customizable parallel optimization opportunities. HPC AI500 V3.0 implements two modules, bagging management (BM) and model parallelism management (MPM), to achieve this customizability. BM determines the algorithm adopted in data sampling and the number of base models. MPM determines the degree of parallelism inside each base model. Through these two modules, users can customize the number of base models and the degree of parallelism to make the trade-off between the model quality and training speed. Based on HPC AI500 [[57](#_bookmark68)], we evaluate HPC AI500 V3.0 on typical HPC systems to show its scalability and customizability.

Our main contributions are summarized as follows:

* + - * According to the unique challenges of HPC AI Benchmarking, we reformulated the HPC AI scalability issue (Section [2](#_bookmark3)).
      * We propose the bagging approach in HPC AI benchmarking to achieve relevancy and scalability and implement HPC AI500 V3.0, a scalable and customizable framework for HPC AI bench- marking (Section [3](#_bookmark9)).

*𝑘 𝑖*∈*𝐵𝑘*

where *𝐵𝑘* ∈ {1*,* 2*,* 3*,* … *, 𝑁* } is the batch sampled from the whole dataset and *𝜂𝑘* is the learning rate of iteration *𝑘*. |*𝐵𝑘*| refers to the batchsize. The ratio of *𝑁* and |*𝐵𝑘*| determines the number of iterations in a training

epoch.

* 1. *The scalability issue*

With the convergence of AI and HPC, both academia and industry players [[63](#_bookmark74)–[65](#_bookmark75)] leverage the computing power of HPC systems to speed up the training process of DL models. However, SGD training has a significant drawback, limited by the batchsize.

* + 1. *The limitation of batchsize*

Although there are millions of data in a DL dataset with the size

*𝑁* [[66](#_bookmark76)], the intrinsic sequential property of SGD only allows a batch with size *𝐵𝑘* (e.g., *𝐵𝑘* = 256) of data to be processed in parallel in

an iteration. We call the computation cost required by a batch as the *Amount of Parallel Computation in an Iteration(in short, APC)*. Compared to Linpack, whose APC can be tuned by the size of the input matrix, the APC of DL workloads is usually a constant and can be represented as:

|∑*𝐵𝑘* |

*𝐴𝑃 𝐶𝑑𝑙* = *𝐶𝑜𝑚𝑝𝑢𝑡𝑎𝑡𝑖𝑜𝑛*(*𝑓𝑗* (*𝑥*)) (3)

*𝑗*=1

where *𝑗* ∈ {1*,* 2*,* 3*,* … *,* |*𝐵𝑘*|} is data that is randomly sampled from the DL dataset with the size *𝑁* and included in batch *𝐵𝑘*. And

*𝐶𝑜𝑚𝑝𝑢𝑡𝑎𝑡𝑖𝑜𝑛*(*𝑓𝑗* (*𝑥*)) is the computation cost required by the DL model

to process a single data and can be measured by FLOPs.

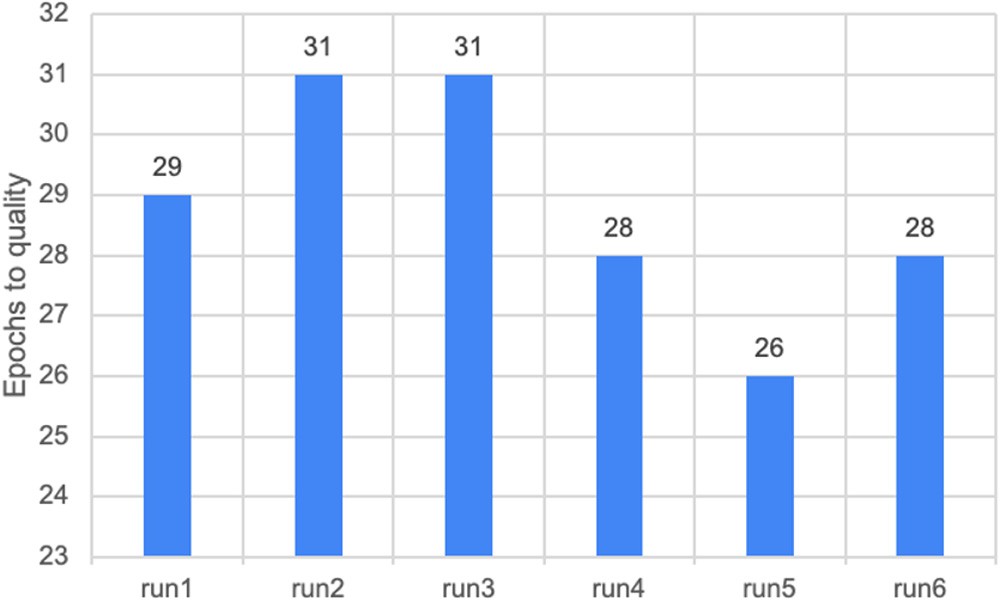
Eq. ([3](#_bookmark5)) indicates that *𝐴𝑃 𝐶𝑑𝑙* is determined by the |*𝐵𝑘*|. However, the value of *𝐵𝑘* is usually a small number, where |*𝐵𝑘*| *≪ 𝑁* . Specif- ically, *𝐵𝑘* ∈ {16*,* 32*,* 64*,* 256*...*512} in many DL applications such as

image classification [[2](#_bookmark28)] and object detection [[4](#_bookmark30),[5](#_bookmark31)]. In this context, it is hard to fully utilize the computing power of HPC systems, which are usually equipped with hundreds or even thousands of nodes. Taking

**Table 1**

Comparison of HPC AI500 V3.0 against HPC AI500 V1.0, V2.0, and other HPC AI benchmarks. The equivalence, affordability, representativeness, and repeatability issues are resolved in our previous work HPC AI500 V2.0 [[57](#_bookmark68)]. HPC AI500 V3.0 is an HPC AI benchmarking framework which inherits and extends HPC AI500 V2.0 with scalability. HPC AI500 V3.0 can naturally integrate other HPC AI benchmarks. ‘‘✗’’ and ‘‘✓’’ indicate whether they have the corresponding properties. ‘‘-’’ indicates not verified.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Related work | Equivalence | Representativeness | Affordability | Repeatability | Scalability |
| HPC AI500 V1.0 (2018) [[50](#_bookmark62)] | ✗ | ✓ | ✓ | ✗ | ✗ |
| HPL-AI (2019) [[55](#_bookmark66)] | ✓ | ✗ | ✓ | ✓ | ✓ |
| Deep500 (2019) [[60](#_bookmark71)] | ✗ | ✗ | ✓ | – | ✗ |
| HPC AI500 V2.0 (2020) [[57](#_bookmark68)] | ✓ | ✓ | ✓ | ✓ | ✗ |
| AIPerf (2020) [[54](#_bookmark65)] | ✓ | ✓ | ✓ | – | ✓ |
| MLPerf (HPC) (2021) [[61](#_bookmark72)] | ✓ | ✓ | ✓ | ✓ | ✗ |
| **HPC AI500 V3.0** | ✓ | ✓ | ✓ | ✓ | ✓ |

the ImageNet/ResNet-50 training on Summit [[50](#_bookmark62),[57](#_bookmark68)] as an example.

|*𝐵𝑘*| = 512, according to Eq. ([3](#_bookmark5)), the APC of ImageNet/ResNet-50 is

11 776 GFLOPs. Considering Summit has 4608 nodes (six Nvidia Tesla

GPUs in each node), each node only can allocate the computation of

11776 = 2*.*55 GFLOPs, which is far away from the peak performance of

4608 [1](#_bookmark7)

six V100 GPUs.

Naively enlarging |*𝐵𝑘*| to improve *𝐴𝑃 𝐶𝑑𝑙* leads to a degradation

proposed in [[36](#_bookmark55),[46](#_bookmark60),[67](#_bookmark77)] indeed increase |*𝐵𝑘*| to a larger number, but it in the model quality due to the sharp minima [[36](#_bookmark55),[46](#_bookmark60),[67](#_bookmark77)]. The tricks

is still far from the peak performance of the HPC system, leading to poor resource utilization. Furthermore, the proposed tricks are empirical, lack generalization ability, and depend on a specific DL workload.

relationship between *𝐵𝑘* and model quality. So far, no research can systematically and theoretically quantify the

* + 1. *The reformulation of HPC AI scalability*

Based on the aforementioned analysis, we reformulate the HPC AI scalability from the following two perspectives. In the previous work [[57](#_bookmark68)], we have discussed how to resolve equivalence, represen- tativeness, affordability, and repeatability issues.

* + - * The *𝐴𝑃 𝐶𝑑𝑙* should be large enough to accommodate the scale

and computing capability of HPC systems. To be specific, it is

necessary to maintain a high resource utilization and near-linear speed up.

creasing the *𝐴𝑃 𝐶𝑑𝑙* and |*𝐵𝑘*|. Otherwise, the whole training pro- • The model quality should be maintained or improved while in-

cess is meaningless.

Compared to the traditional HPC scalability, which focuses on

scalability emphasizes the restraint of model quality and batchsize |*𝐵𝑘*|. scale efficiency and resource utilization [[24](#_bookmark46)], the reformulated HPC AI

* 1. *Prior work*

In addition to the other AI benchmarks [[29](#_bookmark51)–[34](#_bookmark53)], MLPerf (HPC) [[61](#_bookmark72)], HPL-AI [[55](#_bookmark66)], AIPerf [[54](#_bookmark65)], and HPC AI500 [[50](#_bookmark62),[57](#_bookmark68)] are representative HPC AI benchmarking works. Among them, the earliest work is the HPC AI500 V1.0 [[50](#_bookmark62)], dating back to 2018. HPC AI500 V1.0 [[50](#_bookmark62)]

and V2.0 [[57](#_bookmark68)] and MLPerf(HPC) fail to tackle the scalability issue and focus on selecting typical HPC AI applications and parallel-based optimizations. HPL-AI and AIPerf manage to achieve scalability but bring other problems. HPL-AI evaluates HPC systems by performing mixed-precision LU decomposition at the kernel level. Same to HPL,

it can increase the *𝐴𝑃 𝐶* by adjusting the size of the input matrix.

However, LU decomposition is irrelevant to most AI workloads [[57](#_bookmark68)].

The AIPerf methodology is inspired by AutoML, whose core process is performed by NAS. Although AutoML can scale automatically with the number of nodes, the high randomness of NAS ([Fig.](#_bookmark8) [2](#_bookmark8)) calls into question whether AutoML is desirable as an HPC AI benchmark. [Table](#_bookmark6) [1](#_bookmark6) summarizes the related work chronologically and compares our work with other related work in five dimensions.

1 The peak performance of six V100 GPUs in terms of FLOPS is: 6 ×

15*.*7 × 103 GFLOPS = 94*.*2 × 103 GFLOPS.

**/ig. 2.** The randomness of NAS. In different runs, the amount of computation required to train NAS to the target model quality varies, which leads to unfair and unrepeatable evaluation.

# HPC AI500 V3.0

This section first presents the HPC AI500 v3.0 methodology. Then we detail the design, workflow, and customizable configuration. Fi- nally, we introduce the measurement method and the proposed metrics.

* 1. *Methodology*
     1. *Ensemble learning and bagging*

The ensemble learning idea is to solve a common problem by combining the predictions of a group of base models. Rather than making decisions depending on a single model, a group of models makes it possible for ensemble learning to reduce the variance of pre- dictions [[59](#_bookmark70)], so-called the wisdom of crowds [[68](#_bookmark78)]. Bagging (Bootstrap AGGregatING) is a fundamental paradigm of Ensemble learning. As its name suggests, bagging consists of two parts: bootstrapping and aggregating. Bootstrapping is essentially a data sampling process with replacement from the original dataset. The data generated through this process is called the bootstrapped dataset. The training process of bagging is highly parallel as each base model in the ensemble is trained based on its corresponding bootstrapped dataset rather than the original dataset. After finishing the training, the final decision is aggregated by averaging all the predictions of the base models.

* + 1. *Applying bagging in HPC AI benchmarking*

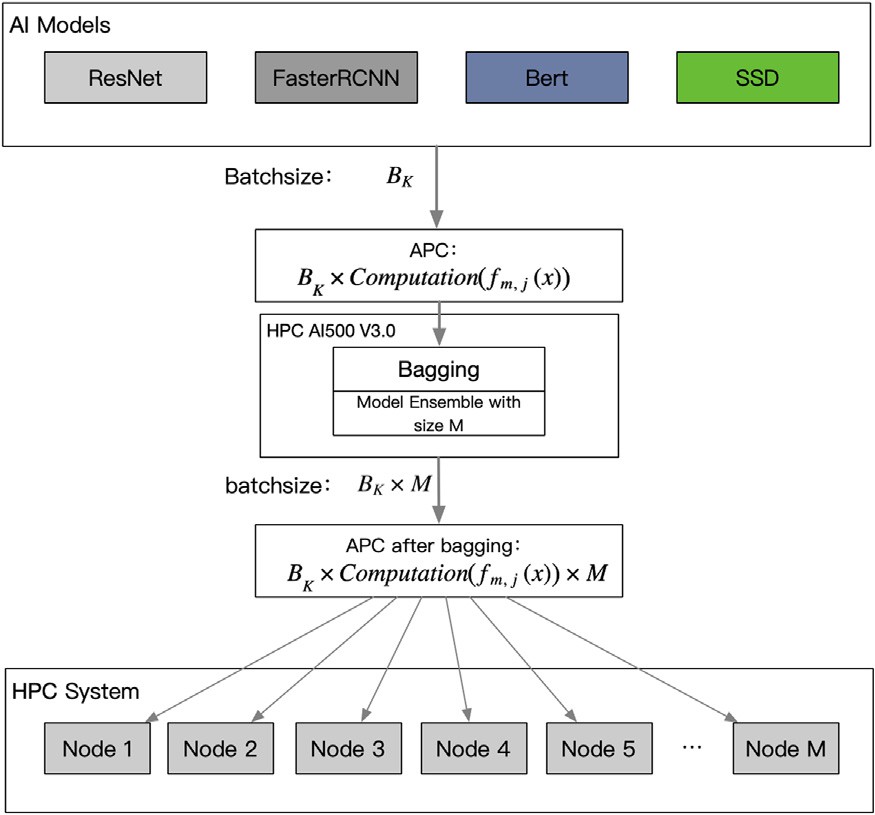
thing is to enlarge the *𝐴𝑃 𝐶* to keep up with the increasingly larger scale For HPC AI benchmarking, to tackle the scalability problem, the first

of HPC systems. Inspired by the Bagging, we introduce *the base model ensemble* on the basis of the training of a single model in the previous AI benchmark like HPC AI500 V2.0. We rewrite Eq. ([3](#_bookmark5)) in the following bagging form:

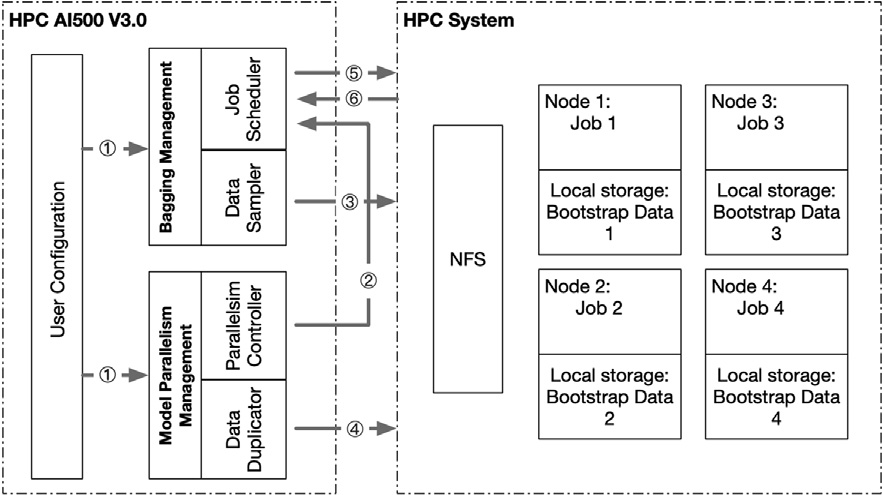
∑*𝑀* |∑*𝐵𝑘* |

*𝐴𝑃 𝐶𝑑𝑙* = *𝐶𝑜𝑚𝑝𝑢𝑡𝑎𝑡𝑖𝑜𝑛*(*𝑓𝑚,𝑗* (*𝑥*)) (4)

*𝑚*=1 *𝑗*=1



**/ig. 3.** The system overview of HPC AI500 V3.0. APC refers to the amount of parallel computing in an iteration.



**/ig. 4.** System design and workflow of HPC AI500 V3.0. NFS refers to the Network File System of HPC systems that each node shares.

where *𝑀* is the number of the base models in the ensemble, *𝑓𝑚* is the

*𝑚𝑡ℎ* base model. Note that each base model is the instance of the orig-

inal model, so the computation cost of each base model is equivalent

to that in Eq. ([3](#_bookmark5)). Compared to AutoML, the re-sampled bootstrapped dataset makes every base model dissimilar, but the computational logic of each model is consistent, guaranteeing no randomness shown in [Fig.](#_bookmark8) [2](#_bookmark8). All the base model in the ensemble is trained independently,

enlarging the |*𝐵𝑘*| by *𝑀* times, and so does *𝐴𝑃 𝐶𝑑𝑙* . Considering each

the ensemble size *𝑀* and the parallelism degree inside a base model base model may train in a distributed manner across several nodes,

*𝑃* \_*𝑑𝑒𝑔𝑟𝑒𝑒* should satisfy Eq. ([5](#_bookmark13)), where *𝑆𝑦𝑠𝑠𝑐𝑎𝑙𝑒* refers to how many

nodes are contained in an HPC system.

*𝑆𝑦𝑠*\_*𝑠𝑐𝑎𝑙𝑒* = *𝑀* × *𝑃* \_*𝑑𝑒𝑔𝑟𝑒𝑒* (5)

* 1. *System overview*

Based on the Bagging approach, we present HPC AI500 V3.0 and the system overview shown in [Fig.](#_bookmark11) [3](#_bookmark11). HPC AI500 V3.0 does not focus on workload selection and construction as previous AI benchmarks [[29](#_bookmark51),[31](#_bookmark52), [34](#_bookmark53)]. Instead, it is a framework that is compatible with these efforts. We briefly introduce the positioning and role of HPC AI500 V3.0 through [Fig.](#_bookmark11) [3](#_bookmark11). This figure shows that HPC AI500 V3.0 scales out the upper-layer AI workloads on lower-layer HPC systems by adaptively increasing

*𝐴𝑃 𝐶𝐴𝐼* . Specifically, the batchsize of each AI workload is initially only

a fixed *𝐵𝑘*. After Bagging, a set of *𝑀* base models are generated, which increases *𝐴𝑃 𝐶𝑑𝑙* by concurrently running *𝑀* base models. This way,

the base model set, *𝑀* , can be adjusted according to the system size, thereby, achieves higher resource utilization. In addition, the size of

corresponding to the same adjustable input matrix size in HPL, to adapt to the future growth of the HPC system scale.

* 1. *System design and workflow*

HPC AI500 V3.0 consists of three components, namely, User Con- figuration (UC), Bagging Management (BM), and Model Parallelism Management (MPM). BM focuses on managing Bagging, including Job

Controller and Data Sampler. Job Controller schedules *𝑀* jobs to the

corresponding nodes, then launch training, and finally aggregates the

predictions. Note that each job corresponds with a base model training. Data Sampler controls the data sampling algorithm. MPM is divided

sets the parallel mode and *𝑃* \_*𝑑𝑒𝑔𝑟𝑒𝑒*. Data Duplicator is responsible for into Parallelism Controller and Data Duplicator. Parallelism Controller

copying and migrating data according to parallelism-related configura- tion. As shown in [Fig.](#_bookmark12) [4](#_bookmark12), we summarize the workflow of HPC AI500 V3.0 as follows:

configurations, including job number, equal to ensemble size *𝑀* , 1. UC sends the configurations to BM and MPM. BM receives the

and saves the DL model and original dataset that needs to be trained. MPM receives the configurations, such as parallelism

mode, *𝑃𝑑𝑒𝑔𝑟𝑒𝑒*, and *𝑆𝑦𝑠𝑠𝑐𝑎𝑙𝑒*.

1. Parallelism Controller in MPM checks if *𝑀* , *𝑃𝑑𝑒𝑔𝑟𝑒𝑒*, and *𝑆𝑦𝑠𝑠𝑐𝑎𝑙𝑒*

according to the received messages (e.g., *𝑇 𝑎𝑠𝑘*1 → *𝑁𝑜𝑑𝑒*1), then satisfy Eq. ([5](#_bookmark13)) and generates the mapping of the jobs to the nodes

sends this mapping to Job Scheduler in BM.

1. Data Sampler in BM determines the sampling algorithm and generates the bootstrap data for each task. All the generated data is sent to the NFS of the HPC system.
2. Data Duplicator in MPM duplicates the bootstrap data according

ple, *𝐽 𝑜𝑏*1− *> 𝑁𝑜𝑑𝑒*1 means the bootstrap data in Job1 only need to the mapping that Parallelism Controller generates. For exam-

**Table 2**

The Customizable Configuration of HPC AI500 V3.0. *𝑁𝑜𝑑𝑒*\_*𝑎𝑐𝑐* refers to the number of

accelerators equipped in a node of the HPC system.

Type Default setting Alternatives

**Table 3**

The FLOPs calculation rules for primary operators in a DL model.

*𝐾* refers to the kernel size, *𝐶𝑖𝑛* and *𝐶𝑜𝑢𝑡* refers to the input and output channel, *𝐻* and *𝑊* refers to the data size, *𝐺𝑟𝑜𝑢𝑝𝑠𝑖𝑧𝑒* refers to the group size of the convolution, and *𝐹 𝐿* refers to the flatten

Basic *𝑃𝑑𝑒𝑔𝑟𝑒𝑒* = *𝑁𝑜𝑑𝑒*\_*𝑎𝑐𝑐*

*𝑀* = *𝑆𝑦𝑠𝑠𝑐𝑎𝑙𝑒*

*𝑃𝑑𝑒𝑔𝑟𝑒𝑒*

Any *𝑀* and *𝑃𝑑𝑒𝑔𝑟𝑒𝑒*

that satisfy Eq. ([5](#_bookmark13))

layer used in the Fully-connected.

Operators FLOPs

Learning Rate Scheduler

warm-up schema and linear scaling [[69](#_bookmark79)]

LARS [[35](#_bookmark54)], LAMB [[70](#_bookmark80)]

Convolution 2 × *𝐾*2 × *𝐶𝑖𝑛* × *𝐻* × *𝑊* × *𝐶𝑜𝑢𝑡*

Optimizer SGD with momentum Adam [[71](#_bookmark81)], AdaGrad [[72](#_bookmark82)]

Depth-wise Convolution 2 × *𝐾*2 × *𝐶𝑖𝑛* × *𝐻* × *𝑊*

Group Convolution 2×*𝐾*2 ×*𝐶𝑖𝑛* ×*𝐻* ×*𝑊* ×*𝐶𝑜𝑢𝑡*

*𝐺𝑟𝑜𝑢𝑝𝑠𝑖𝑧𝑒*

Data Precision for Training

Data Precision

for Communication

FP16 mixed-precision, Int8

FP32 FP16, Int8

Fully-connected *𝐹 𝐿𝑖𝑛* × *𝐹 𝐿𝑜𝑢𝑡*

Element-wise *𝐶𝑜𝑢𝑡* × *𝐻* × *𝑊*

Pooling *𝐶𝑖𝑛* × *𝐻* × *𝑊*

Parallel Mode data parallelism model parallelism, pipeline parallelism [[73](#_bookmark83)],

mixed parallelism

Normalization *𝐶𝑖𝑛* × *𝐻* × *𝑊*

Communication Mode

synchronous all-reduce 2D-Torus [[41](#_bookmark57)], Hierarchical all-reduce [[40](#_bookmark56)]

*3.6. Measurement*

According to Eq. ([4](#_bookmark10)) and Eq. ([6](#_bookmark16)), to determine the *𝐹 𝐿𝑂𝑃 𝑆*, we

Framework TensorFlow [[74](#_bookmark84)] PyTorch [[75](#_bookmark85)],

Mindspore [[76](#_bookmark86)]

to be duplicated once. All the duplicated data is sent to the local storage of the corresponding node.

1. Job Scheduler sends the job to the corresponding nodes and launches the training of the whole ensemble.
2. After the training is finished, Job Scheduler collects all the ensemble output and then makes the final prediction.
   1. *Customizable configuration*

basic configuration, such as *𝑀* and |*𝑃𝑑𝑒𝑔𝑟𝑒𝑒*|, we summarize other In order to maintain the optimization space, in addition to the

customizable configurations in [Table](#_bookmark14) [2](#_bookmark14). We provide a default setting and some alternatives in each configuration type. Note that alternatives just list the favored option, and the user can customize the efficient implementation according to their situation.

* 1. *Metrics*

Same as HPL, we use *FLOPS* (Floating point operations per second) as our primary metric:

∑*𝑁* ∕|*𝐵𝑘* | *𝐴𝑃 𝐶*

need to first measure the *𝐶𝑜𝑚𝑝𝑢𝑡𝑎𝑡𝑖𝑜𝑛*(*𝑓* (*𝑥*)). Although profiling tools

such as Nsight [[77](#_bookmark87)] are able to count the FLOPs by kernel replay, it is dependent on the Nvidia hardware. In order to reduce the influence of the hardware and the hardware-specific optimizations performed by bundled low-level libraries (e.g., CuDnn for Nvidia GPUs), we present an analytical method to calculate the FLOPs that a DL model requires. Modern AI frameworks, such as TensorFlow, describe the computa- tion of a DL model using a directed acyclic graph (DAG) that consists of multiple nodes and edges. The Node in the DAG represents a kind of operator, and the edge represents the data flow. Each operator defines a computation logic and receives the data from the input edge, and then sends the intermediate result to the next operator after finishing its computation. Unlike HPL, which has only one kind of operator (LU decomposition), a DL model usually consists of multiple operators with different kinds. Hence, we summarize the most frequent operators in DL as shown in [Table](#_bookmark15) [3](#_bookmark15). In addition to these listed operators, we ignore other low-proportion operators contained in the DL model. Based on

this table, we can calculate the *𝐶𝑜𝑚𝑝𝑢𝑡𝑎𝑡𝑖𝑜𝑛*(*𝑓* (*𝑥*)) by traversing the

DAG.

*3.7. Implementation details*

Job scheduler of the Bagging management module is based on SLURM (Simple Linux Utility for Resource Management) [[78](#_bookmark88)]. SLRUM is the most commonly used scheduling system in HPC AI systems, fault-tolerant and highly scalable, and suitable for Linux clusters of

*𝐹 𝐿𝑂𝑃 𝑆* = *𝑖*=1 *𝑑𝑙*

*𝑇𝑒𝑝𝑜𝑐ℎ*

(6)

different sizes. We implement the submitted job script based on the

*𝑠𝑏𝑎𝑡𝑐ℎ* interface of SLRUM and use *𝑠𝑖𝑛𝑓 𝑜* and *𝑠𝑚𝑎𝑝* to monitor the

where *𝑇𝑒𝑝𝑜𝑐ℎ* refers to the training time of one epoch and *𝑁* ∕|*𝐵𝑘*| refers

to the number of iterations in one training epoch. In addition to FLOPS,

we also adopt a metric that considers both system throughput and model quality, namely Valid FLOPS (VFLOPS) [[57](#_bookmark68)]. The definition of *VFLOPS* is shown as follows:

*𝑉 𝐹 𝐿𝑂𝑃 𝑆* = *𝐹 𝐿𝑂𝑃 𝑆* ∗ *𝑝𝑒𝑛𝑎𝑙𝑡𝑦*\_*𝑐𝑜𝑒𝑓 𝑓 𝑖𝑐𝑖𝑒𝑛𝑡* (7)

*𝑝𝑒𝑛𝑎𝑙𝑡𝑦*\_*𝑐𝑜𝑒𝑓 𝑓 𝑖𝑐𝑖𝑒𝑛𝑡* = (*𝑎𝑐ℎ𝑖𝑒𝑣𝑒𝑑*\_*𝑞𝑢𝑎𝑙𝑖𝑡𝑦*∕*𝑡𝑎𝑟𝑔𝑒𝑡*\_*𝑞𝑢𝑎𝑙𝑖𝑡𝑦*)*𝑛* (8)

where *𝑝𝑒𝑛𝑎𝑙𝑡𝑦*\_*𝑐𝑜𝑒𝑓 𝑓 𝑖𝑐𝑖𝑒𝑛𝑡* is used to penalize or award the FLOPS based on the achieved quality. *𝑎𝑐ℎ𝑖𝑒𝑣𝑒𝑑*\_*𝑞𝑢𝑎𝑙𝑖𝑡𝑦* refers to the actual model quality achieved in the evaluation. *𝑡𝑎𝑟𝑔𝑒𝑡*\_*𝑞𝑢𝑎𝑙𝑖𝑡𝑦* is predefined in the [Table](#_bookmark21) [4](#_bookmark21). The value of *𝑛* defines the sensitivity to the model quality.

According to the setting of HPC AI500 V2.0 [[57](#_bookmark68)], we set n as 10 for Extreme Weather Analytics and 5 for Image Classification.

training progress of the base model in each job, and the basic unit of job scheduling is a container implemented by Docker [[79](#_bookmark89)]. According

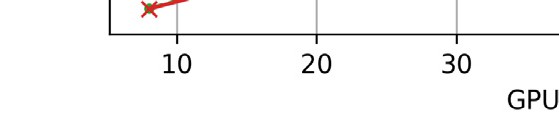
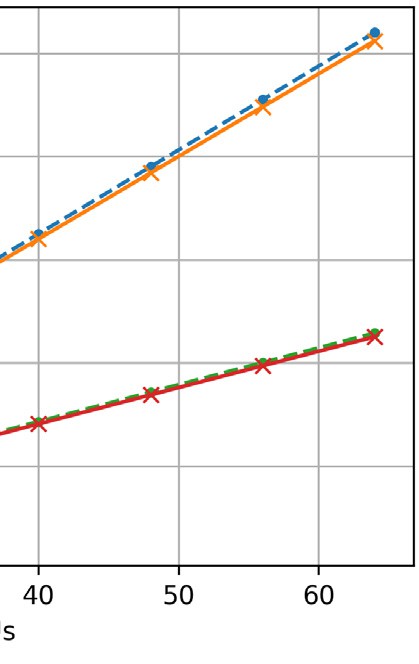
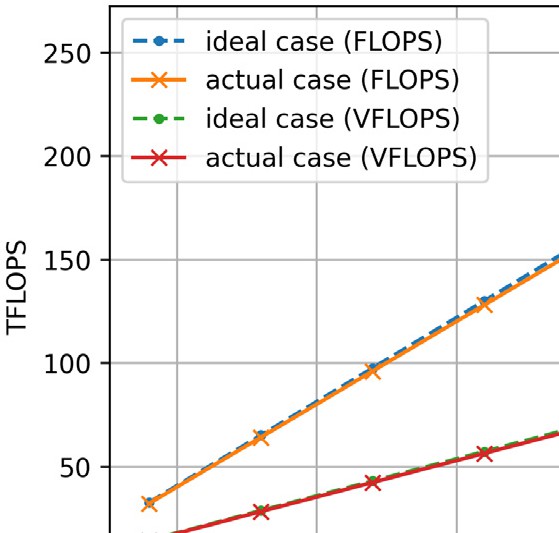
guarantees that the *𝑖𝑡ℎ* training sample is selected *𝑛* (*𝑛* ∈ {0*,* 1*,* 2*...*}) to the literature [[58](#_bookmark69)], the implemented random sampling algorithm

tion of *𝜆* = 1, so the probability of at least one occurrence of the *𝑖𝑡ℎ* times. The probability of the times approximates the Poisson distribu- sample is 1 − ( 1 ) = 0*.*632. So for any Bagging base classifier, about

*𝑒*

36.8% of the samples of the original dataset will not be used at the time of training. The default parallel implementation in the parallel management module uses data parallelism implemented by Horovod and OpenMPI, which is also the most common parallel method in HPC AI systems [[17](#_bookmark39)–[19](#_bookmark41)]. The measurement of bandwidth is divided into intra-node communication and inter-node communication, and we use Nvidia-smi (NVIDIA System Management Interface) tool [[80](#_bookmark90)] to monitor communication within nodes and use iftop tool [[81](#_bookmark91)] to monitor communication between nodes.

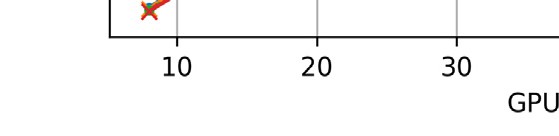
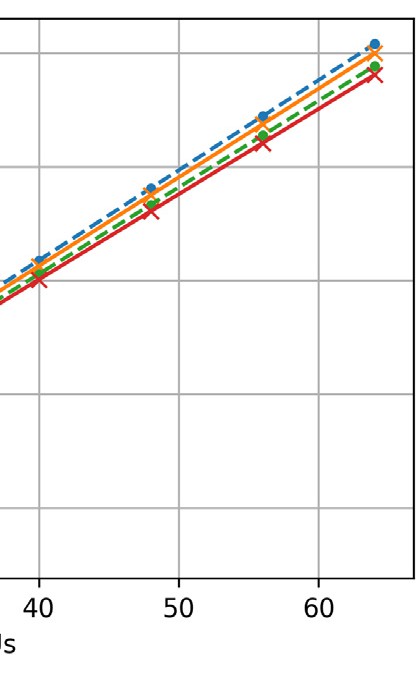
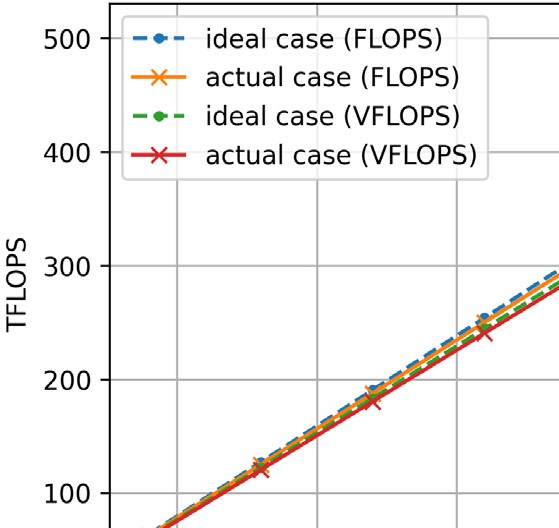
manner. Here, our default implementation is based on HPC AI500 V2.0 [[57](#_bookmark68)], a well-received HPC AI benchmark that mainly consists of two workloads, covering AI applications in business and scientific computing. As shown in [Table](#_bookmark21) [4](#_bookmark21), Image Classification uses ResNet- 50 [[2](#_bookmark28)]and ImageNet [[66](#_bookmark76)] for training, which is a well-known showcase for optimizing HPC AI systems. Extreme Weather Analytics [[82](#_bookmark92)] is a representative scientific application, it uses Faster-RCNN for detecting the extreme weather in the climate image. Each climate image in Extreme Weather Dataset consists of 16 channels and contains four extreme weather patterns.



* 1. *The scalability experiments*

of HPC AI500 V3.0, as shown in [Table](#_bookmark14) [2](#_bookmark14). We set the *𝑃𝑑 𝑒𝑔𝑟𝑒𝑒* = 8, The scalability experiments are conducted with the default setting

which is equal to the number of GPUs in a node. In each node, a job is distributed to 8 GPUs by using data parallelism. We perform the experiment sequentially on different system scales, typically the



*𝑆𝑦𝑠𝑠𝑐𝑎𝑙𝑒* = 8*,* 16*,* 24*,* 32*,* 40*,* 48*,* 56*,* 64 GPUs. According to Eq. ([5](#_bookmark13)), the

corresponding job number is *𝑀* = 1*,* 2*,* 3*,* 4*,* 5*,* 6*,* 7*,* 8. The results of scala-

bility experiments are shown in [Fig.](#_bookmark19) [5](#_bookmark19). As we can see, HPC AI500 V3.0 shows near-linear scalability in both FLOPS and VFLOPS. Note that, in

[Fig.](#_bookmark17) [5(a)](#_bookmark17), the *𝑝𝑒𝑛𝑎𝑙𝑡𝑦*\_*𝑐𝑜𝑒𝑓 𝑓 𝑖𝑐𝑖𝑒𝑛𝑡* = 0*.*44 leads to a gap between the

VFLOPS line and FLOPS of Extreme Weather Analytics. Furthermore, we measure the GPU utilization by Nsight at the scale of 64 GPUs and the result is shown in [Fig.](#_bookmark22) [6](#_bookmark22). Both Extreme Weather Analytics and Image Classification achieve high GPU utilization.

* 1. *The customizability experiments*



The *𝑝𝑒𝑛𝑎𝑙𝑡𝑦*\_*𝑐𝑜𝑒𝑓 𝑓 𝑖𝑐𝑖𝑒𝑛𝑡* is 0.44 for Extreme Weather Analytics and 0.96 for Image **/ig. 5.** The scalability experiments of HPC AI500 V3.0 in terms of FLOPS and VFLOPS.

Classification.

# Evaluation

* 1. *Experimental setup*
     1. *Hardware*

Our experiments are conducted on a 64GPUs-cluster, consisting of eight nodes, each of which is equipped with one Intel(R) Xeon(R) Platinum 8268 CPU and eight NVIDIA Tesla V100 GPUs. Each GPU in the same node has 32 GB HBM memory, connected by NVIDIA NVLink—a high-speed GPU interconnection whose theoretical peak bi-directional bandwidth is 300 GB/s. The nodes are connected with Ethernet networking with a bandwidth of 10 Gb/s. Each node has 1.5 TB system memory and an 8 TB NVMe SSD disk.

* + 1. *Software*

We use TensorFlow v1.14, compiled with CUDA v10.1 and cuDnn v7.6.2 backend. We use Horovod v0.16.4 for synchronous distributed training, compiled with OpenMPI v3.1.4 and NCCL v2.4.8. NCCL is short for the NVIDIA Collective Communications Library, which is a closed-source library of multi-GPU collective communication primitives that are topology-aware.

* 1. *Workloads*

HPC AI500 V3.0 is a benchmarking framework, which means any AI benchmark can be integrated into this framework in a bagging

* + 1. *Trade-off between the model quality and training speed*

case. We set the *𝑀* = 8*,* 4*,* 1 while the corresponding *𝑃𝑑𝑒𝑔𝑟𝑒𝑒* = 8*,* 16*,* 64. To exhibit this trade-off, we take Image Classification as the show-

As shown in [Fig.](#_bookmark23) [7](#_bookmark23), the training speed increases along with a decrease in

*𝑀* . When *𝑀* = 1, the process becomes training a single model through

the whole cluster, achieving the highest training speed. However, since only one model makes decisions in the ensemble, the model quality

suffers about a 3% drop compared to the case of *𝑀* = 8. In practical

scenarios, users can choose appropriate *𝑀* and *𝑃𝑑𝑒𝑔𝑟𝑒𝑒* according to their

training speed and model quality requirements.

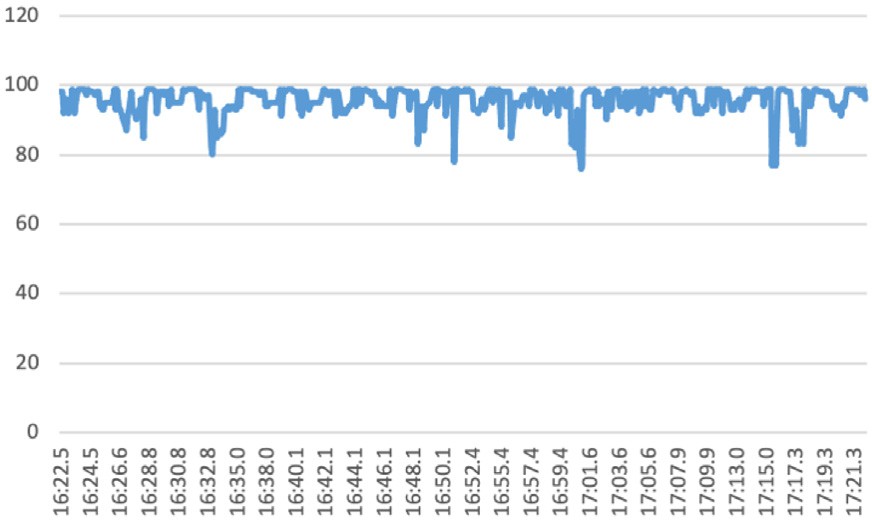
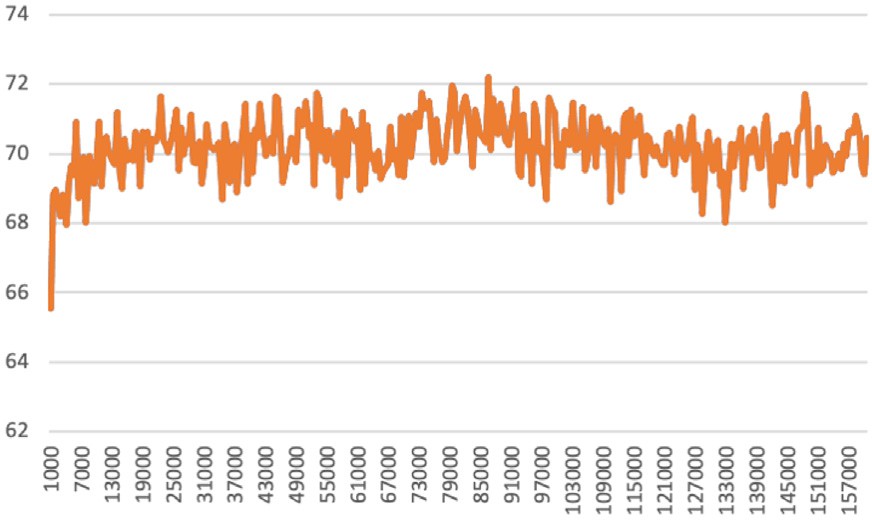
* + 1. *Optimizations*

To show the customizability of HPC AI500, we implement two frequently-used optimizations, mixed-precision training, and commu- nication compression. The former utilizes Tensor Cores in Nvidia Volta architecture to accelerate the model’s fully-connected and convolution layer, allowing a fused-multiply-add computation. When performing mixed precision training with a Tensor Core, we use FP16 for calcu- lation and FP32 for accumulation. The latter is the communication compress-on that compresses the tensor precision for synchronizing from 32FP to 16FP to reduce communication overhead. We configure the optimization experiments in the same way as Section [4.3](#_bookmark18), and the results are shown in [Fig.](#_bookmark24) [8](#_bookmark24). We compared the optimized version to the original version to observe the corresponding effect. Since mixed- precision Extreme Weather Analysis leads to a significant loss of the model quality, here we only report the performance of the model compression. As we can see, mixed-precision training brings about 2x speed up for Image Classification. As for communication compression, it brings about 1.2x for Extreme Weather Analytics but barely has any speed up on Image Classification. The size of the communication tensor in Extreme Weather Analytics is 1.6x larger than that of Image Classification, allowing Extreme Weather Analytics to get a notable benefit.

**Table 4**

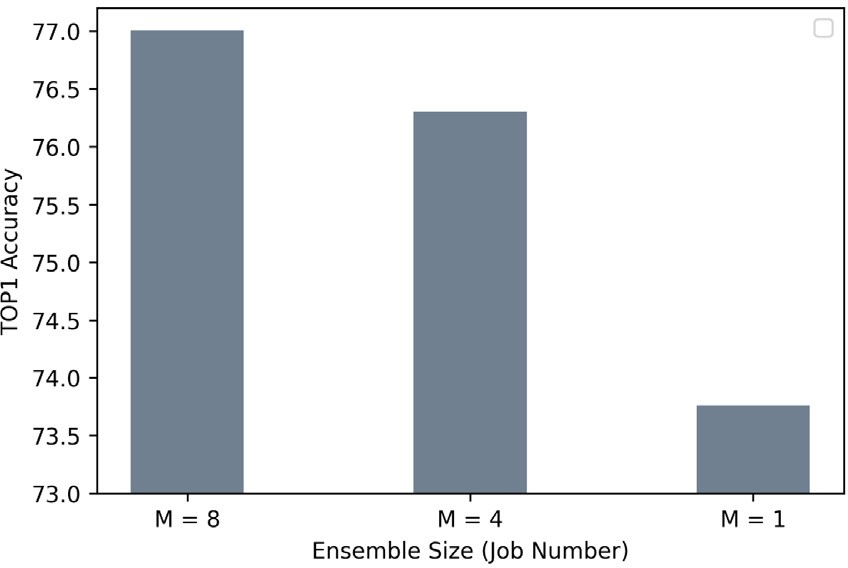
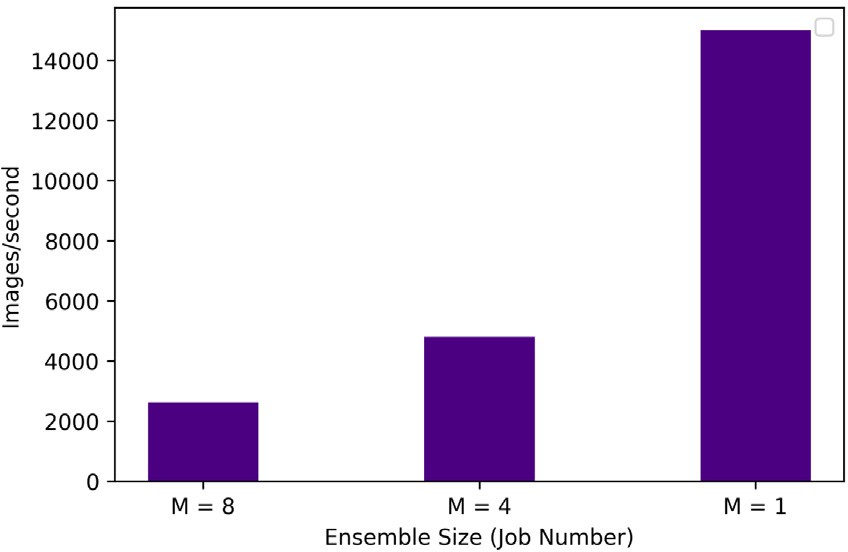
The Specification of HPC AI500 V2.0 workloads [[57](#_bookmark68)]. HPC AI500 V3.0 can integrate any HPC AI benchmarks. In our evaluation, we reuse the HPC AI500 V2.0 workloads for testing.

|  |  |  |  |
| --- | --- | --- | --- |
| Problem domains | Models | Datasets | Target quality |
| Image Classification | ResNet-50 | ImageNet | TOP1 Accuracy  = 0.763 |
| Extreme Weather Analytics | Faster-RCNN | Extreme Weather Dataset | mAP@[IoU=0.5]  = 0.35 |



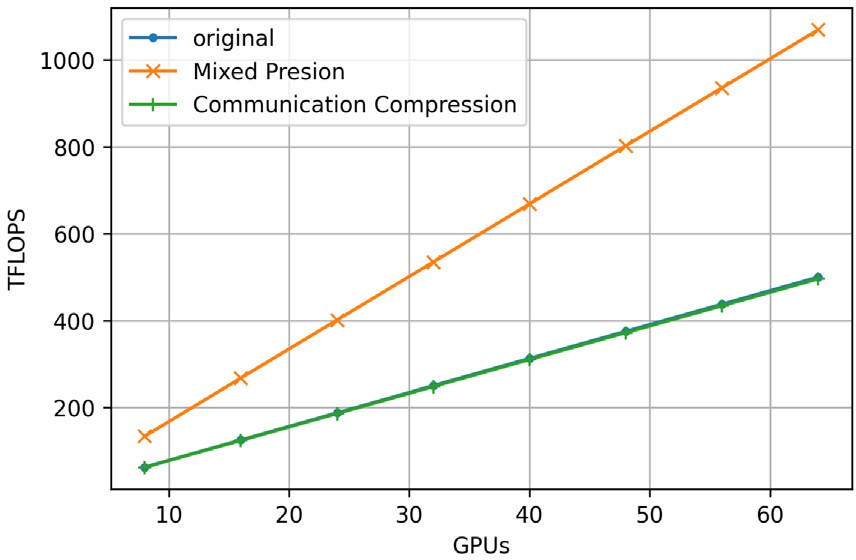
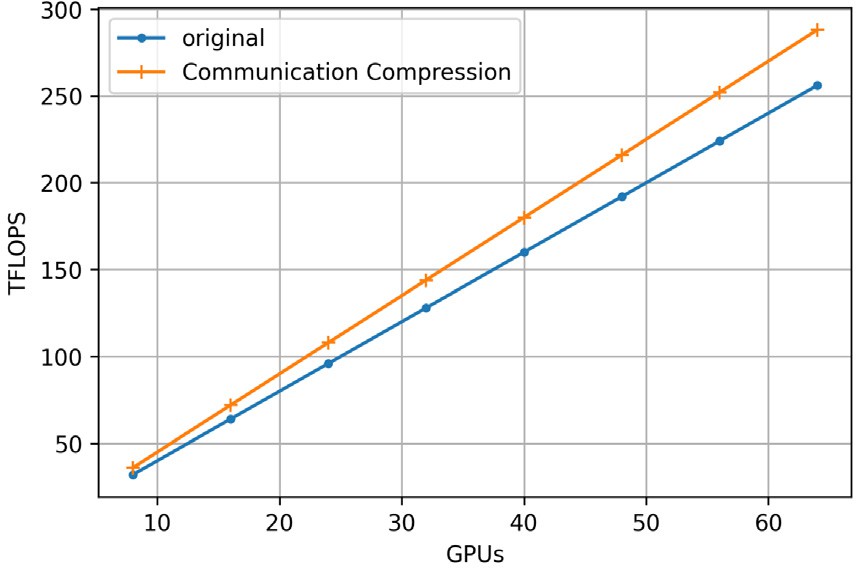


**/ig. 6.** GPU utilization (%) of HPC AI500 V3.0. The *𝑋*-axis represents different time steps.





**/ig. 7.** The trade-off between the training speed and model quality. The workload is Image Classification. We use images per second to indicate how fast the training is.



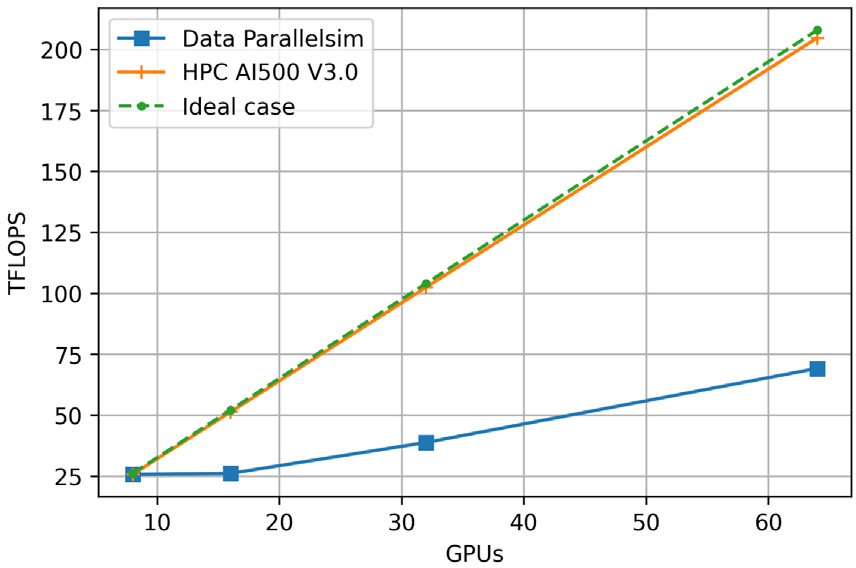
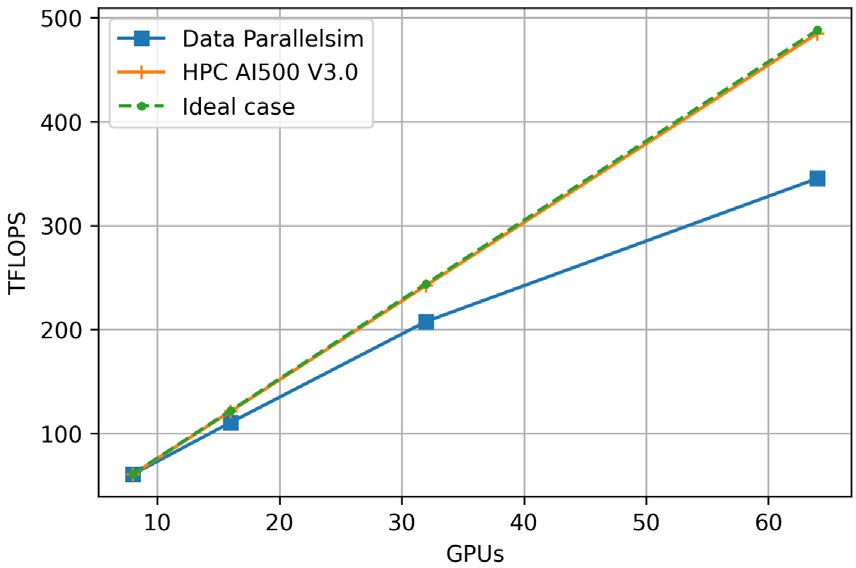


**/ig. 8.** The optimization experiments of HPC AI500 V3.0. In [Fig.](#_bookmark25) [8(b)](#_bookmark25), the lines of the original and mixed precision overlap for their similar performance.

* 1. *Comparison experiments*

We compare our work with data parallelism (DP), which is a main- stream parallel method used in many previous work [[17](#_bookmark39),[18](#_bookmark40),[47](#_bookmark61),[61](#_bookmark72)]. In

this experiment, we focus on scale efficiency in terms of FLOPS. The system scales from 8 GPUs to 64 GPUs. As shown in [Fig.](#_bookmark26) [9](#_bookmark26), the scaling efficiency of DP is much lower than our approach in both Extreme Weather Analysis and Image Classification. The heavy communication



**/ig. 9.** The comparison experiments between HPC AI500 V3.0 against a setting using data parallelism.

overhead of DP is the main reason for this phenomenon because all the model copies of DP need to be synchronized globally at the end of each training step. The base model in the model ensemble of HPC AI500 V3.0 is trained highly independently without synchronization, so the communication overhead is avoided.

# Conclusion

In this paper, we reformulate the HPC AI scalability issue and present HPC AI500 V3.0, a scalable and customizable framework for HPC AI benchmarking. The methodology of HPC AI500 V3.0 allows users to integrate existing AI benchmarks in a bagging manner, a meta-algorithm of ensemble learning with intrinsic high parallelism, leading to scalable benchmarking. The bagging management and model parallelism management of HPC AI500 V3.0 gives users the flexibility to control the size of model ensembles and the degree of model paral- lelism, enabling various optimizations from both system and algorithm levels. Based on HPC AI500 V2.0, which tackles the equivalence, representativeness, affordability, and repeatability issues, HPC AI500 V3.0 provide a complete HPC AI benchmarking framework. Reusing the workloads of HPC AI500 V2.0, we evaluate HPC AI500 V3.0 on a typical HPC system and the experimental results show the scalability and customizability of the proposed benchmarking framework.

# Declaration of competing interest

The authors declare that they have no known competing finan- cial interests or personal relationships that could have appeared to influence the work reported in this paper.

# References

1. [A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb1) [convolutional neural networks, Adv. Neural Inf. Process. Syst. 25 (2012).](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb1)
2. [K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in:](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb2) [Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb2) [2016, pp. 770–778.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb2)
3. [C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb3) [architecture for computer vision, in: Proceedings of the IEEE Conference on](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb3) [Computer Vision and Pattern Recognition, 2016, pp. 2818–2826.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb3)
4. [S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: Towards real-time object](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb4) [detection with region proposal networks, Adv. Neural Inf. Process. Syst. 28](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb4) [(2015).](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb4)
5. [W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A.C. Berg, SSD:](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb5) [Single shot multibox detector, in: European Conference on Computer Vision,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb5) [Springer, 2016, pp. 21–37.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb5)
6. [J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb6) [real-time object detection, in: Proceedings of the IEEE Conference on Computer](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb6) [Vision and Pattern Recognition, 2016, pp. 779–788.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb6)
7. [A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb7)

[I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb7)

1. J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, 2018, arXiv preprint [arXiv:1810.04805](http://arxiv.org/abs/1810.04805).
2. [K. Han, A. Xiao, E. Wu, J. Guo, C. Xu, Y. Wang, Transformer in transformer,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb9) [Adv. Neural Inf. Process. Syst. 34 (2021).](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb9)
3. [Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015)](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb10)

[436–444.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb10)

1. OpenAI, OpenAI: AI and Compute, <https://openai.com/blog/ai-and-compute/>.
2. A. Gholami, Medium: AI and Memory Wall, [https://medium.com/riselab/ai-and-](https://medium.com/riselab/ai-and-memory-wall-2cb4265cb0b8/) [memory-wall-2cb4265cb0b8/](https://medium.com/riselab/ai-and-memory-wall-2cb4265cb0b8/).
3. M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, B. Catanzaro, Megatron- LM: Training multi-billion parameter language models using model parallelism, 2019, arXiv preprint [arXiv:1909.08053](http://arxiv.org/abs/1909.08053).
4. [T. Brown, B. Mann, N. Ryder, M. Subbiah, J.D. Kaplan, P. Dhariwal, A.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb14) [Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb14) [learners, Adv. Neural Inf. Process. Syst. 33 (2020) 1877–1901.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb14)
5. D. Lepikhin, H. Lee, Y. Xu, D. Chen, O. Firat, Y. Huang, M. Krikun, N. Shazeer, Z. Chen, Gshard: Scaling giant models with conditional computation and automatic sharding, 2020, arXiv preprint [arXiv:2006.16668](http://arxiv.org/abs/2006.16668).
6. W. Fedus, B. Zoph, N. Shazeer, Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity, 2021, arXiv preprint [arXiv:2101.03961](http://arxiv.org/abs/2101.03961).
7. [A. Mathuriya, D. Bard, P. Mendygral, L. Meadows, J. Arnemann, L. Shao, S. He,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb17)

[T. Kärnä, D. Moise, S.J. Pennycook, et al., CosmoFlow: Using deep learning](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb17) [to learn the universe at scale, in: SC18: International Conference for High](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb17) [Performance Computing, Networking, Storage and Analysis, IEEE, 2018, pp.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb17) [819–829.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb17)

1. [T. Kurth, S. Treichler, J. Romero, M. Mudigonda, N. Luehr, E. Phillips, A.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb18) [Mahesh, M. Matheson, J. Deslippe, M. Fatica, et al., Exascale deep learning](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb18) [for climate analytics, in: SC18: International Conference for High Performance](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb18) [Computing, Networking, Storage and Analysis, IEEE, 2018, pp. 649–660.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb18)
2. [W. Jia, H. Wang, M. Chen, D. Lu, L. Lin, R. Car, E. Weinan, L. Zhang, Pushing](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb19) [the limit of molecular dynamics with ab initio accuracy to 100 million atoms](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb19) [with machine learning, in: SC20: International Conference for High Performance](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb19) [Computing, Networking, Storage and Analysis, IEEE, 2020, pp. 1–14.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb19)
3. Z. Guo, D. Lu, Y. Yan, S. Hu, R. Liu, G. Tan, N. Sun, W. Jiang, L. Liu, Y. Chen, et al., Extending the limit of molecular dynamics with ab initio accuracy to 10 billion atoms, 2022, arXiv preprint [arXiv:2201.01446](http://arxiv.org/abs/2201.01446).
4. Oak Ridge National Laboratory, Summit, <https://www.olcf.ornl.gov/summit/>.
5. Fujitsu, Fugaku, <https://www.fujitsu.com/global/about/innovation/fugaku/>.
6. [J.L. Hennessy, D.A. Patterson, Computer Architecture: A Quantitative Approach,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb23) [Elsevier, 2011.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb23)
7. [J.J. Dongarra, P. Luszczek, A. Petitet, The LINPACK benchmark: Past, present](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb24) [and future, Concurr. Comput.: Pract. Exper. 15 (9) (2003) 803–820.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb24)
8. J. Dongarra, Top500 Website, <https://www.top500.org/>.
9. J. Dongarra, CM-5 in TOP500 List, [https://www.top500.org/lists/top500/1993/](https://www.top500.org/lists/top500/1993/06/) [06/](https://www.top500.org/lists/top500/1993/06/).
10. J. Dongarra, Fugaku in TOP500 List, [https://www.top500.org/news/japan-](https://www.top500.org/news/japan-captures-top500-crown-arm-powered-supercomputer/) [captures-top500-crown-arm-powered-supercomputer/](https://www.top500.org/news/japan-captures-top500-crown-arm-powered-supercomputer/).
11. J. Zhan, Call for establishing benchmark science and engineering, 2021, arXiv preprint [arXiv:2112.09514](http://arxiv.org/abs/2112.09514).
12. [R. Adolf, S. Rama, B. Reagen, G.-Y. Wei, D. Brooks, Fathom: Reference workloads](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb29) [for modern deep learning methods, in: 2016 IEEE International Symposium on](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb29) [Workload Characterization, IISWC, IEEE, 2016, pp. 1–10.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb29)
13. [C. Coleman, D. Narayanan, D. Kang, T. Zhao, J. Zhang, L. Nardi, P. Bailis,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb30)

[K. Olukotun, C. Ré, M. Zaharia, Dawnbench: An end-to-end deep learning](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb30) [benchmark and competition, Training 100 (101) (2017) 102.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb30)

1. H. Zhu, M. Akrout, B. Zheng, A. Pelegris, A. Phanishayee, B. Schroeder, G. Pekhimenko, TBD: Benchmarking and analyzing deep neural network training, 2018, arXiv preprint [arXiv:1803.06905](http://arxiv.org/abs/1803.06905).
2. W. Gao, F. Tang, L. Wang, J. Zhan, C. Lan, C. Luo, Y. Huang, C. Zheng, J. Dai, Z. Cao, et al., AIBench: An industry standard internet service AI benchmark suite, 2019, arXiv preprint [arXiv:1908.08998](http://arxiv.org/abs/1908.08998).
3. [V.J. Reddi, C. Cheng, D. Kanter, P. Mattson, G. Schmuelling, C.-J. Wu, B.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb33) [Anderson, M. Breughe, M. Charlebois, W. Chou, et al., Mlperf inference bench-](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb33) [mark, in: 2020 ACM/IEEE 47th Annual International Symposium on Computer](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb33) [Architecture, ISCA, IEEE, 2020, pp. 446–459.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb33)
4. [P. Mattson, C. Cheng, G. Diamos, C. Coleman, P. Micikevicius, D. Patterson, H.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb34) [Tang, G.-Y. Wei, P. Bailis, V. Bittorf, et al., Mlperf training benchmark, Proc.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb34) [Mach. Learn. Syst. 2 (2020) 336–349.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb34)
5. [Y. You, Z. Zhang, C.-J. Hsieh, J. Demmel, K. Keutzer, Imagenet training](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb35) [in minutes, in: Proceedings of the 47th International Conference on Parallel](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb35) [Processing, 2018, pp. 1–10.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb35)
6. P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, K. He, Accurate, large minibatch SGD: Training imagenet in 1 hour, 2017, arXiv preprint [arXiv:1706.02677](http://arxiv.org/abs/1706.02677).
7. T. Akiba, S. Suzuki, K. Fukuda, Extremely large minibatch SGD: Training resnet-50 on imagenet in 15 minutes, 2017, arXiv preprint [arXiv:1711.04325](http://arxiv.org/abs/1711.04325).
8. M. Cho, U. Finkler, S. Kumar, D. Kung, V. Saxena, D. Sreedhar, Powerai DDL, 2017, arXiv preprint [arXiv:1708.02188](http://arxiv.org/abs/1708.02188).
9. V. Codreanu, D. Podareanu, V. Saletore, Scale out for large minibatch SGD: Residual network training on ImageNet-1K with improved accuracy and reduced time to train, 2017, arXiv preprint [arXiv:1711.04291](http://arxiv.org/abs/1711.04291).
10. X. Jia, S. Song, W. He, Y. Wang, H. Rong, F. Zhou, L. Xie, Z. Guo, Y. Yang, L. Yu, et al., Highly scalable deep learning training system with mixed-precision: Training imagenet in four minutes, 2018, arXiv preprint [arXiv:1807.11205](http://arxiv.org/abs/1807.11205).
11. H. Mikami, et al., Imagenet/resnet-50 training in 224 seconds, 2018, arXiv preprint [arXiv:1811.05233](http://arxiv.org/abs/1811.05233).
12. C. Ying, S. Kumar, D. Chen, T. Wang, Y. Cheng, Image classification at supercomputer scale, 2018, arXiv preprint [arXiv:1811.06992](http://arxiv.org/abs/1811.06992).
13. M. Yamazaki, A. Kasagi, A. Tabuchi, T. Honda, M. Miwa, N. Fukumoto, T. Tabaru, A. Ike, K. Nakashima, Yet another accelerated SGD: Resnet-50 training on imagenet in 74.7 seconds, 2019, arXiv preprint [arXiv:1903.12650](http://arxiv.org/abs/1903.12650).
14. MLCommons, MLPerf-Training-Result-V1.1, [https://mlcommons.org/en/training-](https://mlcommons.org/en/training-normal-11//) [normal-11//](https://mlcommons.org/en/training-normal-11//).
15. Preferred networks website, <https://www.preferred.jp/en/>.
16. N.S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, P.T.P. Tang, On large- batch training for deep learning: Generalization gap and sharp minima, 2016, arXiv preprint [arXiv:1609.04836](http://arxiv.org/abs/1609.04836).
17. A. Sergeev, M. Del Balso, Horovod: Fast and easy distributed deep learning in TensorFlow, 2018, arXiv preprint [arXiv:1802.05799](http://arxiv.org/abs/1802.05799).
18. [J. Rasley, S. Rajbhandari, O. Ruwase, Y. He, Deepspeed: System optimizations](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb48) [enable training deep learning models with over 100 billion parameters, in:](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb48) [Proceedings of the 26th ACM SIGKDD International Conference on Knowledge](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb48) [Discovery & Data Mining, 2020, pp. 3505–3506.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb48)
19. [N. Shazeer, Y. Cheng, N. Parmar, D. Tran, A. Vaswani, P. Koanantakool, P.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb49) [Hawkins, H. Lee, M. Hong, C. Young, et al., Mesh-tensorflow: Deep learning for](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb49) [supercomputers, Adv. Neural Inf. Process. Syst. 31 (2018).](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb49)
20. [Z. Jiang, W. Gao, L. Wang, X. Xiong, Y. Zhang, X. Wen, C. Luo, H. Ye, X.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb50) [Lu, Y. Zhang, et al., HPC AI500: A benchmark suite for HPC AI systems,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb50) [in: International Symposium on Benchmarking, Measuring and Optimization,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb50) [Springer, 2018, pp. 10–22.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb50)
21. [D. Narayanan, A. Harlap, A. Phanishayee, V. Seshadri, N.R. Devanur, G.R.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb51) [Ganger, P.B. Gibbons, M. Zaharia, PipeDream: Generalized pipeline parallelism](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb51) [for DNN training, in: Proceedings of the 27th ACM Symposium on Operating](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb51) [Systems Principles, 2019, pp. 1–15.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb51)
22. [Z. Jia, M. Zaharia, A. Aiken, Beyond data and model parallelism for deep neural](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb52) [networks, Proc. Mach. Learn. Syst. 1 (2019) 1–13.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb52)
23. data-parallelim, <https://en.wikipedia.org/wiki/Data_parallelism>.
24. [Z. Ren, Y. Liu, T. Shi, L. Xie, Y. Zhou, J. Zhai, Y. Zhang, Y. Zhang, W. Chen,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb54) [AIPerf: Automated machine learning as an AI-HPC benchmark, Big Data Min.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb54) [Anal. 4 (3) (2021) 208–220.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb54)
25. [S. Kudo, K. Nitadori, T. Ina, T. Imamura, Prompt report on exa-scale HPL-](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb55) [AI benchmark, in: 2020 IEEE International Conference on Cluster Computing,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb55) [CLUSTER, IEEE, 2020, pp. 418–419.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb55)
26. B. Zoph, Q.V. Le, Neural architecture search with reinforcement learning, 2016, arXiv preprint [arXiv:1611.01578](http://arxiv.org/abs/1611.01578).
27. [Z. Jiang, W. Gao, F. Tang, L. Wang, X. Xiong, C. Luo, C. Lan, H. Li, J. Zhan,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb57) [HPC AI500 v2. 0: The methodology, tools, and metrics for benchmarking HPC AI](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb57) [systems, in: 2021 IEEE International Conference on Cluster Computing, CLUSTER,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb57) [IEEE, 2021, pp. 47–58.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb57)
28. [L. Breiman, Bagging predictors, Mach. Learn. 24 (2) (1996) 123–140.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb58)
29. [Z.-H. Zhou, Ensemble learning, in: Machine Learning, Springer, 2021, pp.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb59) [181–210.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb59)
30. [T. Ben-Nun, M. Besta, S. Huber, A.N. Ziogas, D. Peter, T. Hoefler, A modular](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb60) [benchmarking infrastructure for high-performance and reproducible deep learn-](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb60) [ing, in: 2019 IEEE International Parallel and Distributed Processing Symposium,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb60) [IPDPS, IEEE, 2019, pp. 66–77.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb60)
31. [S. Farrell, M. Emani, J. Balma, L. Drescher, A. Drozd, A. Fink, G. Fox, D.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb61) [Kanter, T. Kurth, P. Mattson, et al., MLPerf™ HPC: A holistic benchmark suite](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb61) [for scientific machine learning on HPC systems, in: 2021 IEEE/ACM Workshop](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb61) [on Machine Learning in High Performance Computing Environments, MLHPC,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb61) [IEEE, 2021, pp. 33–45.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb61)
32. S. Ruder, An overview of gradient descent optimization algorithms, 2016, arXiv preprint [arXiv:1609.04747](http://arxiv.org/abs/1609.04747).
33. [R. Farber, AI-HPC is Happening Now, InsideHPC Special Report, InsideHPC, LLC,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb63) [2017.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb63)
34. [E.A. Huerta, A. Khan, E. Davis, C. Bushell, W.D. Gropp, D.S. Katz, V. Kindratenko,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb64)

[S. Koric, W.T. Kramer, B. McGinty, et al., Convergence of artificial intelligence](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb64) [and high performance computing on NSF-supported cyberinfrastructure, J. Big](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb64) [Data 7 (1) (2020) 1–12.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb64)

1. [H. Lee, A. Merzky, L. Tan, M. Titov, M. Turilli, D. Alfe, A. Bhati, A. Brace,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb65)

[A. Clyde, P. Coveney, et al., Scalable HPC & AI infrastructure for COVID-19](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb65) [therapeutics, in: Proceedings of the Platform for Advanced Scientific Computing](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb65) [Conference, 2021, pp. 1–13.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb65)

1. [J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb66) [hierarchical image database, in: 2009 IEEE Conference on Computer Vision and](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb66) [Pattern Recognition, Ieee, 2009, pp. 248–255.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb66)
2. [I. Kandel, M. Castelli, The effect of batch size on the generalizability of the](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb67) [convolutional neural networks on a histopathology dataset, ICT Express 6 (4)](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb67) [(2020) 312–315.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb67)
3. [J. Surowiecki, The Wisdom of Crowds, Anchor, 2005.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb68)
4. A. Krizhevsky, One weird trick for parallelizing convolutional neural networks, 2014, arXiv preprint [arXiv:1404.5997](http://arxiv.org/abs/1404.5997).
5. Y. You, J. Li, S. Reddi, J. Hseu, S. Kumar, S. Bhojanapalli, X. Song, J. Demmel,

K. Keutzer, C.-J. Hsieh, Large batch optimization for deep learning: Training bert in 76 minutes, 2019, arXiv preprint [arXiv:1904.00962](http://arxiv.org/abs/1904.00962).

1. D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, arXiv preprint [arXiv:1412.6980](http://arxiv.org/abs/1412.6980).
2. [J. Duchi, E. Hazan, Y. Singer, Adaptive subgradient methods for online learning](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb72) [and stochastic optimization, J. Mach. Learn. Res. 12 (7) (2011).](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb72)
3. [Y. Huang, Y. Cheng, A. Bapna, O. Firat, D. Chen, M. Chen, H. Lee, J. Ngiam,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb73)

[Q.V. Le, Y. Wu, et al., Gpipe: Efficient training of giant neural networks using](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb73) [pipeline parallelism, Adv. Neural Inf. Process. Syst. 32 (2019).](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb73)

1. [M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb74) [Ghemawat, G. Irving, M. Isard, et al., TensorFlow: A system for large-scale](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb74) [machine learning, in: 12th USENIX Symposium on Operating Systems Design](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb74) [and Implementation, OSDI 16, 2016, pp. 265–283.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb74)
2. [A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb75)

[N. Gimelshein, L. Antiga, et al., Pytorch: An imperative style, high-performance](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb75) [deep learning library, Adv. Neural Inf. Process. Syst. 32 (2019).](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb75)

1. Huawei, Mindspore, <https://www.mindspore.cn/>.
2. Nvidia, Nsight system, <https://developer.nvidia.com/nsight-systems>.
3. Lawrence Livermore National Laboratory, SLURM, <https://slurm.schedmd.com/>.
4. [T. Combe, A. Martin, R. Di Pietro, To docker or not to docker: A security](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb79) [perspective, IEEE Cloud Comput. 3 (5) (2016) 54–62.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb79)
5. Nvidia, Nvidia-smi, [https://developer.nvidia.com/nvidia-system-management-](https://developer.nvidia.com/nvidia-system-management-interface) [interface](https://developer.nvidia.com/nvidia-system-management-interface).
6. iftop, <https://en.wikipedia.org/wiki/Iftop>.
7. [E. Racah, C. Beckham, T. Maharaj, S. Ebrahimi Kahou, M. Prabhat, C. Pal,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb82) [Extremeweather: A large-scale climate dataset for semi-supervised detection,](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb82) [localization, and understanding of extreme weather events, Adv. Neural Inf.](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb82) [Process. Syst. 30 (2017).](http://refhub.elsevier.com/S2772-4859(22)00070-9/sb82)