Generic and ML Workloads in an HPC Datacenter: Node Energy, Job Failures, and Node-Job Analysis

Xiaoyu Chu*[1], Daniel Hofstätter*[2], Shashikant Ilager[2], Sacheendra Talluri[1],
Duncan Kampert[3], Damian Podareanu[3], Dmitry Duplyakin[4], Ivona Brandic[2], Alexandru Iosup[1]
[1]Vrije Universiteit, Amsterdam, the Netherlands
[2]TU Wien, Vienna, Austria
[3]SURF, Amsterdam, the Netherlands
[4]National Renewable Energy Laboratory, Colorado, USA
{x.chu, s.talluri, a.iosup}@vu.nl, {daniel.hofstaetter, shashikant.ilager, ivona.brandic}@tuwien.ac.at,
{duncan.kampert, damian.podareanu}@surf.nl, {dmitry.duplyakin}@nrel.gov

***Abstract*—HPC datacenters offer a backbone to the modern digital society. Increasingly, they run Machine Learning (ML) jobs next to generic, compute-intensive workloads, supporting science, business, and other decision-making processes. However, understanding how ML jobs impact the operation of HPC datacenters, relative to generic jobs, remains desirable but understudied. In this work, we leverage long-term operational data, collected from a national-scale production HPC datacenter, and statistically compare how ML and generic jobs can impact the performance, failures, resource utilization, and energy consumption of HPC datacenters. Our study provides key insights, e.g., ML-related power usage causes GPU nodes to run into temperature limitations, median/mean runtime and failure rates are higher for ML jobs than for generic jobs, both ML and generic jobs exhibit highly variable arrival processes and resource demands, significant amounts of energy are spent on unsuccessfully terminating jobs, and concurrent jobs tend to terminate in the same state. We open-source our cleaned-up data traces on Zenodo (https://doi.org/10.5281/zenodo.13685426), and provide our analysis toolkit as software hosted on GitHub (https://github.com/atlarge-research/2024-icpads-hpc-workload-characterization). This study offers multiple benefits for data center administrators, who can improve operational efficiency, and for researchers, who can further improve system designs, scheduling techniques, etc.**

***Index Terms*—Energy Consumption, Failure Analysis, Cross Analysis, Multivariate Analysis, Machine Learning, GPU, Workload Characterization, System Modeling, HPC, Datacenters.**

I. Introduction

High Performance Computing (HPC) datacenters are important Information and Communications Technology (ICT) infrastructures for our society, particularly for scientific research and its many applications. Contemporary HPC datacenters are well-designed and highly tuned for reliable and resource-efficient execution of CPU-based scientific computing workloads [1]–[4]. However, as HPC datacenters are increasingly hosting Machine Learning (ML) jobs, understanding the different requirements and usage characteristics of such jobs is important for the design and tuning of future HPC datacenters. Early studies identify new energy [5] and failure [6] patterns; existing schedulers and workload managers (e.g., SLURM) do not take into account the unique needs of these new types of jobs [3], [7], potentially leading to substantial waste of researchers' time, and computing energy resources [8]–[10]. Addressing the challenges of comparing ML and generic workloads in HPC datacenters, in this study, we thoroughly investigate node energy, job failures, and joint node-job analysis. In this process, addressing the further challenge of data scarcity from HPC datacenters hosting both ML and generic HPC workloads, we collect and open-source long-term job and node data from a production, national-scale HPC datacenter. Our work leads to various findings and actionable insights, and eventually to stronger capabilities to improve resource allocation policies and job management strategies for combined ML and generic HPC workloads [11], [12].

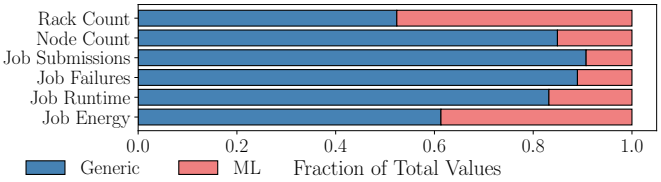


Fig. 1: Generic vs. ML hardware and workload, summary. Energy demands of ML jobs are proportionally higher than their share of submissions and runtime.

It is important to systematically characterize ML and generic workloads running in HPC systems, to understand resource usage behavior, failure patterns, distributions, and correlations across different dependent parameters. Previous studies on HPC trace analysis have primarily focused on job-level data [13]–[15] or machine-level data [16], without considering the interplay between them. However, this can lead to inaccurate insights, as the performance of HPC jobs is highly influenced by the machine and infrastructure conditions [13], [17]–[19]. This can be evidenced by the exemplary results from our analysis in Figure 1, which shows ML jobs running on GPU-accelerated nodes consume 39% of the datacenter's total energy, even though they make up only 15% of the datacenter's nodes and only about 9% of the workload

*Equal contributions, joint first authors.

TABLE I: Cluster overview, shows the total size of the cluster in our study and typical per-node configurations.

	#Nodes	#CPUs	#CPU Cores	CPU TDP	Memory	Storage	#GPUs	GPU TDP	GPU Memory
CPU-only Nodes	287	1	16	125 W	96 GB	1.7 TB	n/a	n/a	n/a
GPU Nodes	51	2	24	210 W	192 GB	2.4 TB	4	1,120 W	96 GB
Total Values	338	489	5872	51,425 W	46,336 GB	644.1 TB	198	53,040 W	3,712 GB

TABLE II: Rack-level overview, shows the size and typical per-rack configurations.

	#Racks	#Nodes	CPU TDP	GPU TDP
CPU-only Racks	11	32	4,000 W	n/a
GPU Racks	10	5	1,050 W	5,600 W

submissions. In addition, ML workloads experience slightly more failures than their node-count indicates.

The objective of this paper is to present an in-depth HPC datacenter analysis and study the characteristics of generic and ML jobs using operational logs. We collected and open-sourced detailed, long-term, job and node-level monitoring information from a production, national-level HPC datacenter—in total, approximately 94 million tuples with 100 metrics, covering four overlapped months of job and node data. We first performed data cleaning and preprocessing (integration) steps, followed by a detailed statistical analysis. We used various methods of data analysis, such as basic statistics, temporal patterns (e.g., trend analysis), distributions (e.g., Probability Density Function (PDF), or Cumulative Distribution Function (CDF)), and Pearson correlations. Our key contributions are:

1) We propose a data processing and characterization method (Section III) for comparing generic and ML workloads, offering insights into large-scale infrastructure by integrating long-term, high-quality node and job data.
2) We unveil how hardware and workloads differ in a heterogeneous HPC environment. Therefore, we study the cluster hardware utilization (Section IV), analyze the characteristics and failure patterns of generic and ML jobs (Section V), and investigate energy usage and correlations among generic and ML job types (Section VI).
3) We contribute to open-science by publishing job and node monitoring data from a relevant HPC datacenter (https://doi.org/10.5281/zenodo.13685426) [20] and the analysis software toolkit (https://github.com/atlarge-research/2024-icpads-hpc-workload-characterization), ensuring reproducibility and supporting further research.

II. System Background

SURF Lisa is a Dutch national-scale datacenter consisting of 338 nodes distributed across 21 racks. Universities and researchers use it for different jobs, including bags of tasks, workflows, and ML training jobs. The jobs are submitted to a SLURM scheduler which then schedules them onto the nodes of the HPC cluster. A job can use a single node or multiple nodes. GPU nodes handle ML workloads for the vast majority of jobs (over 90%), as indicated by the libraries (e.g., *torch*, *cuda*) used by each job, identified via *XALT* by system administrators. Our nodes employ various *Second Generation Intel Xeon* processor models, *NVIDIA TITAN RTX* or *NVIDIA GeForce GTX 1080 Ti* GPUs, all installed in *Dell EMC PowerEdge T640* node enclosures. Typical node configurations can be found in Table I.

In our HPC datacenter, each rack comprises multiple nodes of the same type, i.e., CPU-only racks and GPU racks. A rack can host up to 32 CPU-only (generic) nodes or 7 GPU (ML) nodes, with the most common configurations listed in Table II. The rack air cooling is designed for a 5,500 W capacity. Noteworthy, while CPU-only racks remain within this cooling limit (CPU TDP), GPU racks (CPU+GPU TDPs) often exceed it due to the GPUs' high power demands.

III. Characterization Method

We propose a data-driven characterization method for analyzing and comparing generic and ML workloads, built mainly upon (1) hardware utilization and energy usage, (2) job failures and resource allocation patterns, and (3) joint analysis of node and job metrics. We also take a novel approach by correlating job exit states among concurrently running jobs.

A. Data Collection

We collected approximately 10 months of job data from SLURM, spanning from the end of December 2021 to November 2022. Each job data point is sampled upon its termination, with information on resource allocation, runtime, and exit state. Additionally, we collected roughly 5 months of node data from Prometheus, ranging from June 2022 to November 2022. The sample interval for node data is set at 30s. The node dataset encompasses various software metrics such as packets received and I/O requests, alongside hardware metrics like CPU/GPU power and temperature. The low sampling interval and the large number of metrics presented in this dataset offer the potential for more in-depth and innovative insights into the operations of datacenters [3], [21], which we explore in this study.

B. Processing Metrics and Integrating Datasets

We aggregated raw JSON entries in each row by taking minimum, maximum, mean, and sum values to generate new attributes, e.g., the sum of all GPU power measurements per node and timestamp. Our cleaned node dataset has roughly 128 million tuples in total, with each having 91 features in it.

We integrated job data (Table III-(a)) and node data (Table III-(b)) to enable correlation and energy analysis across both levels. Therefore, we matched each job to corresponding node

TABLE III: Data preparation overview. This work uses 3 different datasets. Legend: #M=Number of metrics, #R=Number of rows in millions, #S=Size of dataset.

ID	Name	Source	Start	End	#M	#R	#S	Description
(a)	Job Dataset	SLURM	2021-12-26	2022-11-01	9	1.60 M	26 MB	ID, dates, node types, #nodes, #cores, state
(b)	Node Dataset	Prometheus	2022-06-30	2022-11-22	91	127.83 M	16 GB	Node memory, network, power usage, etc.
(c)	Joint Dataset	(a) join (b)	2022-06-30	2022-11-01	100	93.95 M	10 GB	Information per-node and related jobs

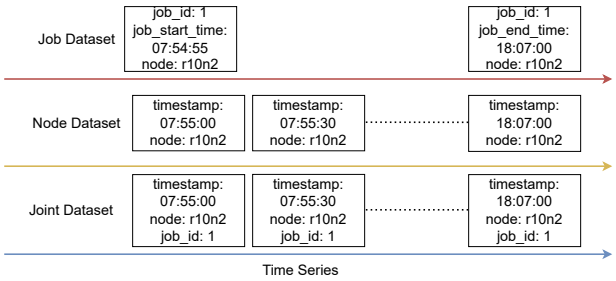


Fig. 2: An example of the data integration process. We match each job record to the fine granular 30s-interval timestamps of the node dataset.

data logs over the job's duration. A similar approach is also taken in [22] to calculate job energy usage utilizing node power metrics. Figure 2 illustrates an example of the data integration process. First, we take a job, e.g., job (job_id: 1) which is executing on r10n2 (node) from 07:54:55 (job_start_time) to 18:07:00 (job_end_time). Second, for all node data samples within this period, the metrics for the job (job_id: 1) are combined with the node data metrics. Afterward, we obtained the combined dataset (Table III-(c)). This method naturally leaves out jobs under 30 seconds that fall between node timestamps, but since they account for less than 0.1% of the total runtime, their impact on energy consumption is negligible, which we mainly discuss by utilizing this combined dataset.

IV. Analysis of Node Utilization and Energy Usage

In this section, we begin by analyzing patterns in overall cluster utilization using the node dataset (TableIII-(b)).

Main Finding 1: GPU nodes under-utilize their CPU (median of 9.6% in *Node Load 1* metric). Both CPU and GPU memory are rarely fully utilized. GPU temperature limits are reached regularly, and their temperatures are affected by hardware topology.
Actionable Insight 1: The CPU-GPU imbalance suggests that operators should provision imbalanced nodes with mixed jobs and save costs by adapting CPU configurations for GPU-heavy workloads. GPU performance can be improved by prioritizing the resource allocation of GPUs at positions with better cooling.

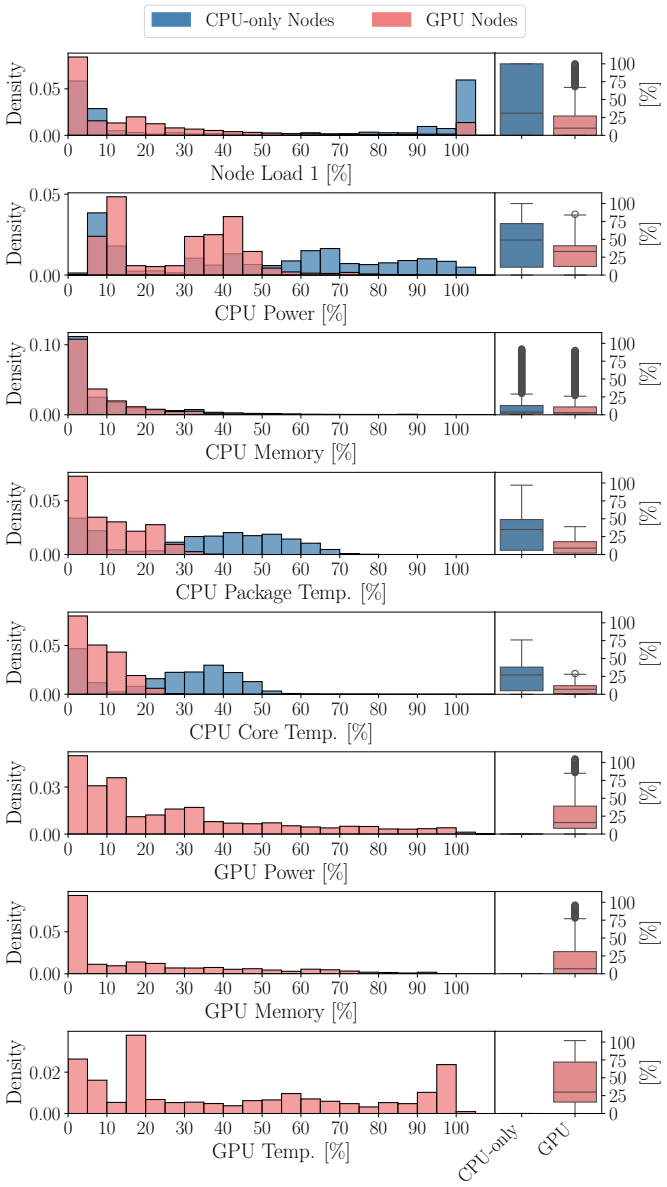


Fig. 3: Normalized node utilization across various metrics is depicted using probability density functions (left) and box plots (right), revealing high GPU temperatures.

A. Node Utilization

Figure 3 shows the probability distribution of various node attributes, normalized to a utilization value. Here, 100% utilization is the maximum value according to hardware specifi-

TABLE IV: Mean GPU power utilization.

GPU Name	GPU0	GPU1	GPU2	GPU3
Mean Power Utilization	34.27%	25.06%	28.02%	23.43%

cations, e.g., the TDP and memory configurations from Table I, and the maximum allowed temperatures according to the CPU and GPU manufacturers. The *Node Load 1* metric reflects the rolling average CPU thread load over the last minute. We consider values equal to or larger than the CPU core count as 100% utilization. We further clip all utilization values to their 99.99th percentile values, to reduce the impact of a few extreme outliers.

Observation 1: CPUs and GPUs under-utilize memory, with mean CPU memory usage below 11% in both CPU-only and GPU nodes and mean VRAM (for GPUs) usage below 19%.

CPU-only nodes show a higher thread load than GPU nodes (see Figure 3). CPU memory (RAM) is mostly under-utilized, with means of 10.3% and 8.1% for CPU-only and GPU nodes, respectively. While RAM is often under-utilized in datacenters [9], [21], [23], [24], it may also be a bottleneck in other cases [25]. The mean GPU memory (VRAM) utilization of 18.7% is higher, but still considerably low. However, low memory utilization may alone not be a sufficient metric for making a statement on memory over-provisioning, since peak loads also have to be handled [26], as evidenced by the many outliers towards 100% memory utilization in the box plots of Figure 3.

Observation 2: GPUs reach temperature limitations regularly. 17% of the time GPU temperature utilization exceeds 90%.

Regarding CPU energy consumption, CPU-only nodes utilize the CPU more often at 100% power than GPU nodes. Consequently, CPU package temperatures are also overall higher for CPU-only nodes. Individual CPU core temperatures show the same pattern and mostly follow the behavior of the CPU package. Due to higher allowed temperatures for individual cores of 101 °C, the lower threshold for package temperatures of 77-87 °C is reached earlier in most cases, meaning throttling of individual cores due to local hot spots is not an issue here.

Due to high GPU power utilization and over-provisioning of GPU TDPs in most GPU racks (Table II), the limited cooling cannot keep up. As a result, GPU temperature utilization regularly reaches 100% and is over 90% about 17.4% of the time. This raises the concern of thermal throttling, which limits GPU performance. One way of dealing with this issue is power-capping GPUs [3], which not only helps to control power surges and temperatures but also reduces energy bills.

B. Temperature Behavior of GPUs

Observation 3: GPU temperatures can vary significantly depending on their position inside the node, with differences of around 9% in temperatures at 100% power utilization.

Looking further into the temperature limitations of GPUs,

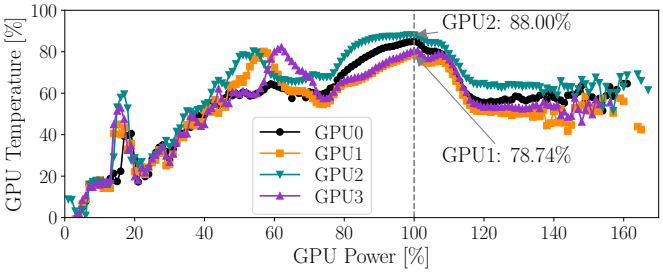


Fig. 4: Average GPU temperature at various power utilizations across GPU indices (0 to 3) in the node. For the same power usage, GPU temperatures vary greatly.

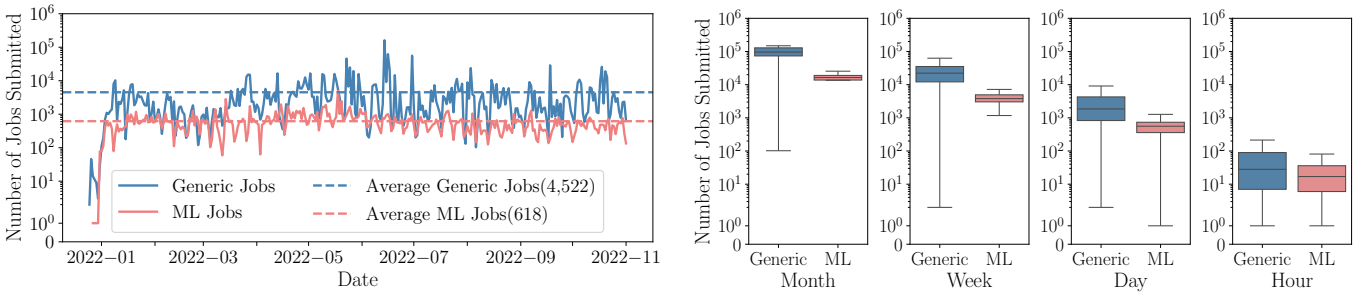
Figure 4 shows how hardware topology affects GPU temperatures, aggregated over 48 GPU nodes equipped with 4 GPUs, where GPUs are indexed according to their physical arrangement inside the node. At 100% power utilization (TDP of GPU), we observed significant differences in GPU temperatures, where GPU2 runs on average about 9% hotter than GPU1, showing that the position of the GPU inside the node has a major influence on thermals. An explanation can be found by looking at the mechanical design of the used GPU models [27], [28] and node enclosures [29]. Inside the node, GPUs are pairwise next to each other, potentially causing their fans to be partially obstructed and taking in hot air from the neighboring GPUs. This heat re-circulation effect has already been a discussed problem at the datacenter level [30]. Here, we evidenced similar phenomena in multi-GPU nodes. Position-dependent thermal behavior of GPUs is also observed in [6], however, their study focused on water-cooled systems, while our work deals with air-cooled environments.

These insights indicate potential opportunities for optimizing GPU performance. Table IV reveals GPU0 is most utilized throughout the cluster. However, GPU1 and GPU3 would be better suited for higher workloads due to their superior cooling. Since GPUs can throttle due to temperature limitations, strategically assigning tasks to cooler GPUs could enhance performance. This can be achieved through, e.g., ML-based scheduling approaches that predict temperature [30].

V. Analysis of Job Characteristics and Failures

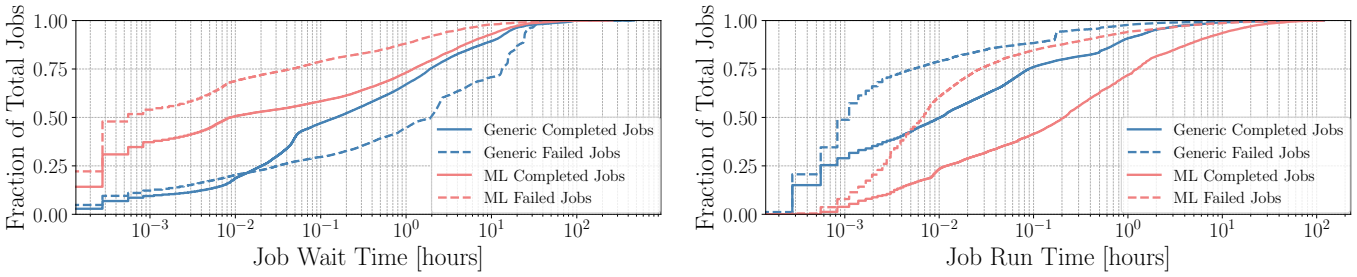
In this section, utilizing the job dataset (TableIII-(a)), we delve into the characterizations of job arrival times, wait times, run times, temporal patterns, and job sizes.

Main Finding 2: ML jobs have longer duration (median of 6.48 minutes) and smaller job sizes (average of 6.81 CPU cores) compared to generic jobs.
Actionable Insight 2: ML jobs' longer duration can be exploited to increase resource utilization with sophisticated scheduling & predictive dynamic optimizations (e.g., turning nodes on/off). Smaller ML job sizes enable specialized network topologies & placement algorithms that could avoid full bandwidth bisection [31].



(a) Jobs submitted by date. **(b) Jobs submissions, aggregated by different time granularity.**

Fig. 5: The total number of submitted generic jobs and ML jobs, showing high variability over time.



(a) Job wait time, CDF plot. **(b) Job run time, CDF plot.**

Fig. 6: Job wait time and run time, showing ML jobs wait shorter but run longer than generic jobs.

A. Job Arrivals

Observation 4: Arrival and demand of generic and ML jobs are highly variable. The number of submitted jobs per day varies by up to three orders of magnitude for both.

Figure 5a gives a timeline overview of submitted generic and ML jobs based on dataset TableIII-(a). The number of submitted jobs per day is highly variable, as the curves are going up and down significantly between days for the 10 months. On average, 4,522 generic jobs and 618 ML jobs are submitted daily. Figure 5b provides the distributions of job submissions across four types of time granularity, ranging from month to hour, excluding outliers based on the three-sigma (3σ) rule. While the median of hourly job submissions shows no significant difference, the maximum number of generic jobs exceeds that of ML jobs by a considerable margin. The daily count of both generic and ML jobs varies by up to three orders of magnitude. However, the number of job submissions is more steady for ML jobs compared to generic ones. The distribution can give insight into the datacenter simulator configuration [32]. Compared to the analysis results of the job data collected in 2020 from a similar study [21], the average amount of ML jobs increased from 320 to 618, reflecting the upward trend of ML research and application.

B. Job Wait Time and Run Time

Observation 5: ML jobs have longer running times (2.71h) and shorter waiting times (1.84h) on average compared to generic jobs (0.83h and 4.21h, respectively).

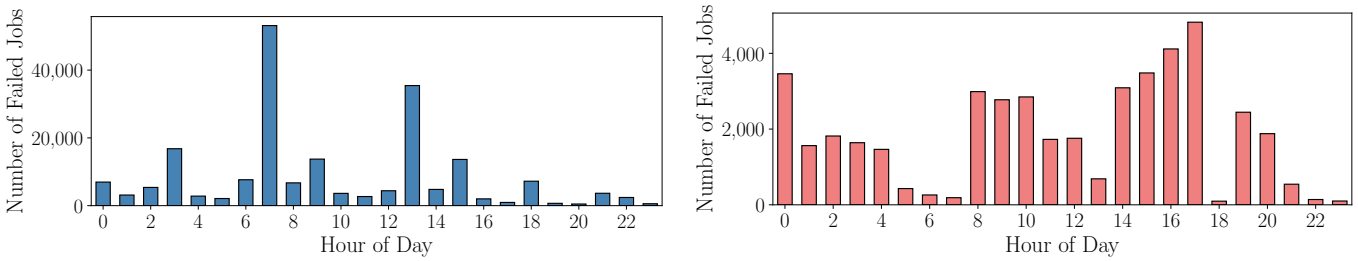
We inspect the wait time and run time of generic and ML jobs, as shown in Figure 6. Overall, the median waiting time for ML jobs (11.00 seconds) is less than for generic jobs (11.65 minutes). Additionally, completed jobs (7.93 minutes) have significantly shorter median waiting times than failed jobs (47.08 minutes). According to a study in [15] on the Argonne Leadership Computing Facility supercomputers, the median waiting time was approximately 1 hour in 2018, while in our cluster, it is 8.2 minutes. The majority of generic jobs (64.67%) and ML jobs (79.83%) wait at most 1 hour.

Overall, the median running time for ML jobs (6.48 minutes) is about 16 times longer than for generic jobs (24 seconds). In contrast, Li et al. [24] report only a roughly 2-fold increase in median runtimes for GPU jobs compared to CPU jobs. Additionally, in our cluster, completed jobs (44 seconds) have longer median running times than failed jobs (5 seconds). Around 85% of ML jobs failed within 6 minutes and 94% failed within 1 hour. 90.80% of generic jobs are usually completed within 1 hour, whereas completed ML jobs have longer durations, with approximately 71.89% taking at most 1 hour. Amvrosiadis et al. [8] report that 80% of jobs have durations shorter than 12 minutes in their analysis of a Google trace, but significantly longer durations of 2-6 hours in 3 other traces they investigated. In contrast, the jobs in our cluster are shorter than 19 minutes at the same percentile, indicating that our job runtimes are between their extremes.

C. Temporal Patterns of Job Failures

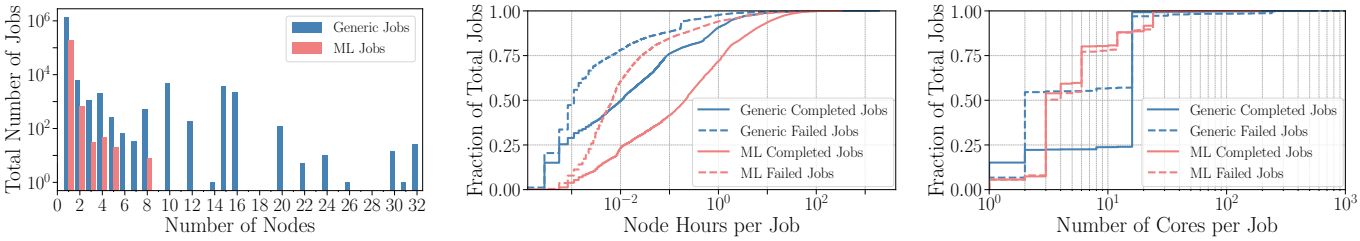
Observation 6: ML job failures have a diurnal pattern, whereas generic job failures exhibit irregular fail behavior, with anomaly peaks on certain days and hours.

To investigate the temporal patterns of generic and ML failed jobs, we aggregated job failures by the hour of the



(a) Generic job failures. **(b) ML job failures.**

Fig. 7: The total number of failed jobs by hour of the day. Generic jobs show higher irregularities than ML jobs.



(a) The number of nodes used per job. **(b) Node hours used per job.** **(c) CPU cores used per job.**

Fig. 8: Job size in nodes, node hours, and CPU cores. Generic jobs utilize more nodes and CPU cores than ML jobs.

day, as shown in Figure 7. Failures in ML jobs exhibit a daily pattern, typically occurring between 8 am and 5 pm. In contrast, generic job failures are irregular and erratic, with occasional spikes on specific days and hours. The daily patterns of ML job failures are similar to previous works [6], [8], [33]. However, the pattern is inconsistent for generic job failures, which may be caused by our datacenter specific operational conditions.

D. Job Size in Nodes, Node Hours, and Cores

Observation 7: ML jobs typically utilize fewer than 8 nodes, whereas generic jobs can utilize a variable number of nodes ranging from 1 to 32.

In Figure 8 we present the size of different jobs in terms of the number of nodes, node hours, and CPU cores. Figure 8a shows the distribution of node allocation per job. The majority of generic jobs (98.50%) and ML jobs (99.60%) only use a single node in the examined cluster. Despite that, generic jobs can utilize a variable number of nodes, ranging up to 32, while ML jobs utilize at most 8 nodes. For both types of jobs, the utilization of multiple nodes is around 1%. Based on Figure 8b, ML jobs use more median node hour time (6.5 minutes) than generic jobs (0.4 minutes), this is majorly due to the longer running time of ML jobs. Because most jobs only use one node, the squashed areas of node hours align with the result of job running time from Figure 6b.

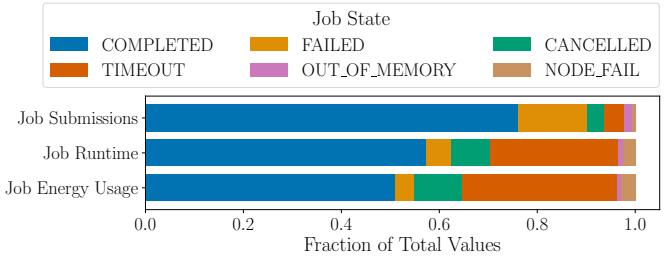
Observation 8: Completed generic jobs typically request more CPU core resources than failed ones, with both surpassing ML jobs in resource demand.

We also inspect the CPU cores used per job. Figure 8c shows the CDF plot of user requests for CPU cores in successful and failed jobs. On average, generic jobs utilize more CPU cores (13.07) compared to ML jobs (6.81), which is expected since ML jobs mainly rely on GPUs for their computation. At the same time, there is no significant distinction between completed jobs (12.67) and failed jobs (10.99). The number of failed generic jobs sharply increases at 2 and 16 cores, indicating that 48% and 40% generic jobs failed at these core counts, respectively. A peak is also observed where 75% completed generic jobs utilized 16 cores. We conjecture most users request one full node via the job scheduler (as indicated in Table I, CPU-only nodes typically have 16 CPU cores). A similar pattern is also observed in earlier works [8], [21]. However, failed and completed ML jobs have an unusual distribution of allocated CPUs, with nearly half (42% and 46% respectively) utilizing 3 cores. This is because the smaller ML jobs are provisioned 3 cores from a 24-core GPU node by the scheduler, confirmed by our datacenter operators.

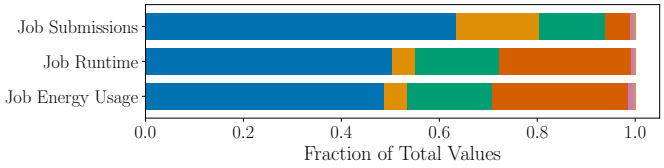
VI. Joint Analysis of Job and Node Data

The combined dataset (TableIII-(c)) enables diverse cross-metric analyses between job and node traces, including energy consumption and correlation analysis.

Main Finding 3: Unsuccessful jobs consume about half of the total cluster energy. Concurrent jobs on the same node show correlations for terminating in the same state, especially generic jobs.
Actionable Insight 3: Checkpointing [34] should be used to save partial work to avoid energy wastage due to failures. Understanding job exit state correlations can enhance failure prediction mechanisms.



(a) Generic Jobs.



(b) ML Jobs.

Fig. 9: Fraction of job states, generic and ML jobs. About half the cluster-wide energy is spent on uncompleted jobs.

TABLE V: Job metrics distribution cluster-wide.

Type	Job Submissions	Job Runtime	Job Energy
Generic	90.72%	83.18%	61.32%
ML	9.28%	16.82%	38.68%

A. Job Submissions, Runtime, and Energy Usage

Observation 9: ML jobs have a smaller share of job submissions (9%) and runtime (17%) on the cluster but contribute to a relatively larger total energy footprint (39%).

Table V provides a cluster-wide overview of the distribution of generic and ML jobs. Despite generic jobs accounting for 91% of the total submissions and 83% of the cumulative runtime, ML jobs demand a disproportionately higher amount of energy, consuming 39% of the cluster-wide energy budget, while generic jobs account for 61%.

Examining the distribution of job termination states within each type of job in Figure 9 reveals some more details. Completed jobs make up the majority of submitted jobs, with a fraction of 76% in Figure 9a for generic ones, and 63% in Figure 9b for ML types. The fraction of submitted jobs failing is smaller for the generic type with 14% than ML with 17%. The biggest difference can be found in the fractions of canceled jobs, which are roughly 4% and 13% for submitted generic and ML jobs, respectively. The overall distribution of submitted generic jobs' exit states aligns with the findings of [22] within a few percentage points. However, they diverge noticeably when comparing them to our spread of ML job states, further emphasizing the importance of looking at ML workloads separately.

Moving on to the sum of job runtimes, job states' proportions shift noticeably. Runtimes for jobs ending in a timeout state take up significant fractions of around 26-27% for both generic and ML types, at the cost of smaller fractions for completed and failed jobs, with similar results being evidenced in the work of [21]. The trend of ML jobs having a relatively higher proportion of canceled jobs seen for job submissions continues for job runtimes. Interestingly, the share of runtime used for jobs exiting with the node failure state is 2.5% among generic jobs, over 10 times higher than the 0.2% among ML jobs.

Observation 10: About 50% of the total cluster energy is used for jobs terminating unsuccessfully.

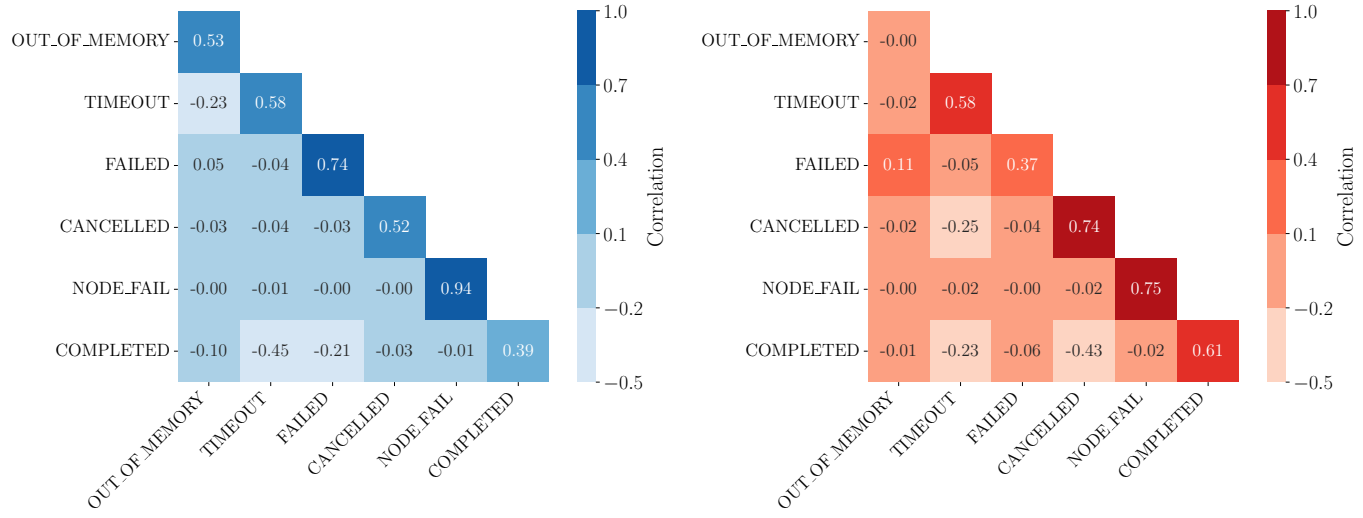
Cumulative energy usage of jobs grouped by state shows very similar patterns to their runtimes, with the fraction of timed-out jobs growing even larger. One key insight from Figure 9 is that even though the majority of submitted jobs are completed successfully, about half of the used energy is spent on jobs resulting in unsuccessful terminations, like failures, timeouts, out-of-memory, or node failures. This highlights the huge potential for energy savings, e.g., by analyzing jobs more intensively before submission or implementing early-stopping mechanisms for long-running jobs that end up in a timeout.

B. Correlation of Job States

Observation 11: High correlations exist between identical job states, indicating that jobs running concurrently on the same node tend to end in the same states. The highest correlations exist for the 'NODE_FAIL' state, with values of 0.94 and 0.75, for generic and ML jobs, respectively.

We investigate whether concurrent jobs fail simultaneously by utilizing the combined dataset to correlate the exit states of two concurrently running jobs on the same node at the same timestamp. This novel analysis checks whether the job exit state correlates with that of other concurrent jobs using Pearson's correlation. Naive usage of Pearson correlation can result in a spurious high correlation, as all the low usage periods could correlate with low job periods. To avoid this, we first find all periods with high power load using peak detection [35], by filtering the dataset for node power usages one standard deviation above the mean. We then correlate the termination states of concurrent jobs across all timestamps.

Figure 10 gives heatmaps for correlations between different job states, split up by generic and ML jobs. In general, there are high correlation coefficients between the same job states, as illustrated on the diagonals of the plots, meaning concurrent jobs are more likely to end up in the same state rather than different ones. However, this observation does not always hold. While generic jobs do have a high correlation of 0.53 for the 'OUT_OF_MEMORY' state in Figure 10a, ML jobs in Figure 10b show no correlation at all. One potential reason for this divergence can be the higher memory demand of generic jobs, compared to ML jobs. Job failures are also more highly correlated among generic jobs than ML jobs, with a correlation of 0.74 compared to 0.37. Correlations are commonly high for generic and ML jobs for the states 'TIMEOUT' and 'CANCELLED', with correlation values above 0.5. 'COMPLETED' jobs show a higher correlation among ML



(a) Generic Jobs. **(b) ML Jobs.**

Fig. 10: Job exit state correlation under high node load. Concurrent jobs tend to terminate in the same exit state.

jobs than generic ones. The strongest correlations between different states are negative with a value of -0.45 between the states 'COMPLETED' and 'TIMEOUT' for generic jobs, and a value of -0.43 between 'COMPLETED' and 'CANCELLED' for ML jobs.

One potential reason for the observed correlations could be that concurrent jobs are submitted by the same users. However, the job dataset lacks user information due to privacy constraints, limiting further investigation of this hypothesis.

VII. Scope and Threats to Validity

Although this study offers new insights into the characterization of generic and ML workloads, we acknowledge that our findings are limited to the specific system and data we used. The threats to the validity include: (1) *Generalizability*: This study characterizes traces from only one datacenter. This is broadly taken in other research works (see Table VI) because of limited access to publicly available data with similar granularities and timelines. Therefore, we use different strategies such as providing a guide to the characterization method, and comparing similar or different findings in other research works. These are commonly used techniques in literature [3], [6], [8], [21]. (2) *Causality*: It is unlikely this kind of study alone can uncover causal relationships between the variables found to be correlated, but it provides corroborating evidence that such correlations exist and thus enables future root-cause analysis. (3) *Categorization*: In our work, we are limited by the knowledge about the application domains of our ML workloads, hence, we look at ML jobs as a single category. While this categorization is sufficient for many types of characterization, it potentially hides other detailed findings. Consequently, dividing ML jobs into multiple distinct subclasses can help to mitigate this issue, such as grouping ML jobs by science domains [36], types [7], and maturity [3].

VIII. Related Work

We briefly summarize comparable related work in Table VI. Relatively, ours is the first to intensively compare generic and ML workloads, covering broadly analyzed metrics, such as utilization, energy, and failures, to the less commonly studied joint characterization of hardware and workload traces.

Most HPC datacenter studies primarily focus on generic workloads [1], [5], [6], [8], [9], [15], [17], [25], [26], [30], [37]–[39], without splitting either jobs types (generic vs ML) or node types (CPU-only vs GPU nodes), as we do in our study. Moreover, we also carry out joint analysis on the combined job and node data, while most other works do not consider this interplay, investigating job and node data independently [1], [7]–[9], [15], [21], [26], [30], [33], [36]–[38]. Most of the studies, focus their analysis on a single datacenter (see Table VI). Works that do analyze multiple datacenters together mainly focus on hardware utilization [5], [8], [15], [38], or evaluate hardware and workload traces separately [1], [8], [15], [38], and commonly do not holistically compare generic and ML based workloads [1], [5], [8], [15], [38].

Shin et al. [6] discuss utilization, the impact of GPU placement on power/temperatures, failure characterization, and cross-analysis of job and node data. However, they do not distinguish between generic and ML workloads like we do in our work. Similar to our work, [22] conducts various workload characterizations, concerning job exit status, time, and energy, while also categorizing jobs into the CPU-only and CPU+GPU classes. Still, they do not show overall hardware utilization, which we add to complement our workload analysis. Previous analysis of our datacenter [21] also looked at generic and ML workloads, but not in a joint job-node fashion as we did. Li et al. [3] focus more on characterizing ML workloads and GPU-accelerated hardware, with less emphasis on comparison with generic workloads or job failure analysis. The work of [24] contrasts GPU and CPU workloads but skips energy and failure analyses, which we include in our study.

TABLE VI: Comparison of related works. Legend: #DC=The number of datacenter systems, #U=Utilization analysis, #E=Energy analysis, #F=Failures analysis, #J=Joint analysis on the relation between node and job data.

Year	Work	Characteristics	Data Scope			Job Types		Characterization Types			
			#DC	Job	Node	Generic	ML	#U	#E	#F	#J
2013	[17]	Characterizing node failures and related factors on a large-scale HPC.	1	✓	✓	✓	✗	✓	✓	✓	✓
2014	[26]	Evaluating job packing in four different metrics.	1	✓	✗	✓	✗	✓	✗	✗	✗
2017	[1]	Comparing failure characteristics of multiple largescale HPC systems.	5	✗	✓	✓	✗	✗	✗	✓	✗
2017	[37]	Characterization and prediction of cloud VM workloads.	1	✓	✗	✓	✗	✓	✗	✓	✗
2018	[38]	HPC workload characterization focus on job geometry and groups.	3	✓	✗	✓	✗	✓	✗	✗	✗
2018	[8]	Characterization of diverse cluster workloads and its impact on research.	4	✓	✗	✓	✗	✓	✗	✓	✗
2019	[25]	Resource efficiency limitation analysis of Alibaba datacenter traces.	1	✓	✓	✓	✗	✓	✗	✗	✓
2020	[39]	HPC job characterization/identification at leadership computing facility.	1	✓	✓	✓	✗	✓	✗	✓	✓
2020	[15]	Long-term analysis of job characteristics on large-scale systems.	2	✓	✗	✓	✗	✓	✗	✗	✗
2020	[5]	Power consumption behavior analysis of jobs on HPC clusters.	2	✓	✓	✓	✗	✓	✓	✗	✓
2020	[30]	Thermal prediction for efficient energy management of clouds using ML.	1	✗	✓	✓	✗	✓	✓	✗	✗
2021	[6]	Power/energy/thermal analysis of a 200PF pre-exascale supercomputer.	1	✓	✓	✓	✗	✓	✓	✓	✓
2021	[36]	Characterizing machine learning I/O workloads on leadership-scale HPC.	1	✓	✗	✓	✓	✗	✗	✗	✗
2022	[3]	AI-workflow classification and analysis on GPU-accelerated systems.	1	✓	✓	✓	✓	✓	✓	✗	✓
2022	[7]	MLaaS characterization on a large-scale heterogeneous GPU-cluster.	1	✓	✓	✗	✓	✓	✗	✗	✗
2023	[33]	Job failure analysis of datacenter with mixed generic/ML workload.	1	✓	✗	✓	✓	✗	✗	✓	✗
2023	[21]	Holistic characterization of both generic and ML job and node data.	1	✓	✓	✓	✓	✓	✓	✓	✗
2023	[9]	Statistical driven datacenter workload analysis of energy and temperature.	1	✗	✓	✓	✗	✓	✓	✗	✗
2023	[22]	Large-scale HPC job power consumption dataset construction and analysis.	1	✓	✓	✓	✓	✗	✓	✓	✓
2023	[24]	Analyzing resource utilization in a heterogeneous large-scale HPC system.	1	✓	✓	✓	✓	✓	✗	✗	✓
2024	**Our work**	**Generic/ML workloads, utilization, energy, failures, and joint analysis.**	**1**	✓	✓	✓	✓	✓	✓	✓	✓

IX. Conclusion

In this work, we identified the emerging challenge of understanding ML workloads in contrast to general HPC workloads. We collected and released job-level and node-level data from a relevant HPC datacenter. Integrating job- and node-data sources, we analyzed utilization, energy, and failure occurrence, and also conducted a joint analysis to reveal the relation between job and node metrics. Our statistical characterization led to 11 major observations, contributing to 3 major findings and 3 actionable insights. Our findings help understand the impact of ever-growing ML jobs on HPC datacenters and provide valuable insights for datacenter operators. We released our datasets and software as open-access artifacts to encourage further research.

Acknowledgment

We thank the Dutch National Supercomputing Center SURF for providing the data. We thank the China Scholarship Council (CSC) for supporting Xiaoyu Chu. We thank the support of Netherlands-funded projects NWO OffSense and GFP 6G FNS, and EU-funded projects MCSA-RISE Cloudstars and Horizon Graph-Massivizer. This research has been partially funded through the projects: High-Performance Integrated Quantum Computing (HPQC), Austrian Research Promotion Agency (FFG) # 897481; Transprecise Edge Computing (Triton), Austrian Science Fund (FWF), DOI: 10.55776/P36870; and Trustworthy and Sustainable Code Offloading (Themis), Austrian Science Fund (FWF), DOI: 10.55776/PAT1668223.

References

[1] S. Gupta, T. Patel, C. Engelmann, and D. Tiwari, "Failures in large scale systems: long-term measurement, analysis, and implications," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC 2017), Denver, CO, USA, November 12 - 17, 2017*, p. 44, ACM, 2017.

[2] B. Schroeder and G. A. Gibson, "A large-scale study of failures in high-performance computing systems," *IEEE Transactions on Dependable and Secure Computing (TDSC 2010)*, vol. 7, no. 4, pp. 337–351, 2010.

[3] B. Li, R. Arora, S. Samsi, T. Patel, W. Arcand, D. Bestor, C. Byun, R. B. Roy, B. Bergeron, J. Holodnak, *et al.*, "Ai-enabling workloads on large-scale gpu-accelerated system: characterization, opportunities, and implications," in *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA 2022)*, pp. 1224–1237, IEEE, 2022.

[4] T. Patel and D. Tiwari, "Perq: Fair and efficient power management of power-constrained large-scale computing systems," in *Proceedings of the 28th International Symposium on High-Performance Parallel and Distributed Computing (HPDC 2019)*, pp. 171–182, 2019.

[5] 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 *2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS 2020)*, pp. 799–809, 2020.

[6] W. Shin, V. Oles, A. M. Karimi, J. A. Ellis, and F. Wang, "Revealing power, energy and thermal dynamics of a 200pf pre-exascale supercomputer," in *International Conference for High Performance Computing, Networking, Storage and Analysis (SC 2021), St. Louis, Missouri, USA, November 14-19, 2021*, p. 12, ACM, 2021.

[7] Q. Weng, W. Xiao, Y. Yu, W. Wang, C. Wang, J. He, Y. Li, L. Zhang, W. Lin, and Y. Ding, "Mlaas in the wild: Workload analysis and scheduling in large-scale heterogeneous gpu clusters," in *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 2022)*, pp. 945–960, 2022.

[8] G. Amvrosiadis, J. W. Park, G. R. Ganger, G. A. Gibson, E. Baseman, and N. DeBardeleben, "On the diversity of cluster workloads and its impact on research results," in *2018 USENIX Annual Technical Conference (USENIX ATC 2018), Boston, MA, USA, July 11-13, 2018*, pp. 533–546, USENIX Association, 2018.

[9] S. Ilager, A. N. Toosi, M. R. Jha, I. Brandic, and R. Buyya, "A data-driven analysis of a cloud data center: Statistical characterization of workload, energy and temperature," in *IEEE/ACM 16th International Conference on Utility and Cloud Computing (UCC 2023)*, 2023.

[10] J. You, J.-W. Chung, and M. Chowdhury, "Zeus: Understanding and optimizing GPU energy consumption of DNN training," in *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 2023), Boston, MA, USA*, pp. 119–139, USENIX Association, Apr. 2023.

[11] A. Qiao, S. K. Choe, S. J. Subramanya, W. Neiswanger, Q. Ho, H. Zhang, G. R. Ganger, and E. P. Xing, "Pollux: Co-adaptive cluster scheduling for goodput-optimized deep learning," in *15th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2021), July 14-16, 2021*, USENIX Association, 2021.

[12] J. Mohan, A. Phanishayee, J. Kulkarni, and V. Chidambaram, "Looking beyond gpus for DNN scheduling on multi-tenant clusters," in *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2022), Carlsbad, CA, USA, July 11-13, 2022*, pp. 579–596, USENIX Association, 2022.

[13] A. K. Paul, O. Faaland, A. Moody, E. Gonsiorowski, K. M. Mohror, and A. R. Butt, "Understanding HPC application I/O behavior using system level statistics," in *27th IEEE International Conference on High Performance Computing, Data, and Analytics (HiPC 2020), Pune, India, December 16-19, 2020*, pp. 202–211, IEEE, 2020.

[14] P. H. Carns, R. Latham, R. B. Ross, K. Iskra, S. Lang, and K. Riley, "24/7 characterization of petascale I/O workloads," in *Proceedings of the 2009 IEEE International Conference on Cluster Computing (CLUSTER 2009), August 31 - September 4, 2009, New Orleans, Louisiana, USA*, pp. 1–10, IEEE Computer Society, 2009.

[15] T. Patel, Z. Liu, R. Kettimuthu, P. Rich, W. E. Allcock, and D. Tiwari, "Job characteristics on large-scale systems: long-term analysis, quantification, and implications," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC 2020), Virtual Event / Atlanta, Georgia, USA, November 9-19, 2020*, p. 84, IEEE/ACM, 2020.

[16] D. Tiwari, S. Gupta, J. Rogers, D. Maxwell, P. Rech, S. Vazhkudai, D. Oliveira, D. Londo, N. DeBardeleben, P. Navaux, *et al.*, "Understanding gpu errors on large-scale hpc systems and the implications for system design and operation," in *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA 2015)*, pp. 331–342, IEEE, 2015.

[17] N. El-Sayed and B. Schroeder, "Reading between the lines of failure logs: Understanding how HPC systems fail," in *2013 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2013), Budapest, Hungary, June 24-27, 2013*, pp. 1–12, IEEE Computer Society, 2013.

[18] M. Snir, R. W. Wisniewski, J. A. Abraham, S. V. Adve, S. Bagchi, P. Balaji, J. F. Belak, P. Bose, F. Cappello, B. Carlson, *et al.*, "Addressing failures in exascale computing," *The International Journal of High Performance Computing Applications*, vol. 28, no. 2, pp. 129–173, 2014.

[19] J. P. White, M. Innus, M. D. Jones, R. L. DeLeon, N. Simakov, J. T. Palmer, S. M. Gallo, T. R. Furlani, M. Showerman, R. Brunner, *et al.*, "Challenges of workload analysis on large hpc systems: a case study on ncsa blue waters," in *Proceedings of the Practice and Experience in Advanced Research Computing 2017 on Sustainability, Success and Impact*, pp. 1–8, 2017.

[20] X. Chu, D. Hofstätter, S. Ilager, S. Talluri, D. Kampert, D. Podareanu, D. Duplyakin, I. Brandic, and A. Iosup, "Generic and ML workloads in an HPC datacenter [Dataset]," Sept. 2024. https://doi.org/10.5281/zenodo.13685426.

[21] L. Versluis, M. Çetin, C. Greeven, K. Laursen, D. Podareanu, V. Codreanu, A. Uta, and A. Iosup, "Less is not more: We need rich datasets to explore," *Future Generation Computer Systems (FGCS 2023)*, vol. 142, pp. 117–130, 2023.

[22] F. Antici, M. S. Ardebili, A. Bartolini, and Z. Kiziltan, "PM100: A job power consumption dataset of a large-scale production HPC system," in *Proceedings of the SC '23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis (SC-W 2023), Denver, CO, USA, November 12-17, 2023*, pp. 1812–1819, ACM, 2023.

[23] I. B. Peng, R. Pearce, and M. B. Gokhale, "On the memory underutilization: Exploring disaggregated memory on HPC systems," in *32nd IEEE International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD 2020), Porto, Portugal, September 9-11, 2020*, pp. 183–190, IEEE, 2020.

[24] J. Li, G. Michelogiannakis, B. Cook, D. Cooray, and Y. Chen, "Analyzing resource utilization in an hpc system: A case study of nersc's perlmutter," in *High Performance Computing*, pp. 297–316, Springer Nature Switzerland, 2023.

[25] J. Guo, Z. Chang, S. Wang, H. Ding, Y. Feng, L. Mao, and Y. Bao, "Who limits the resource efficiency of my datacenter: an analysis of alibaba datacenter traces," in *Proceedings of the International Symposium on Quality of Service (IWQoS 2019), Phoenix, AZ, USA, June 24-25, 2019*, pp. 39:1–39:10, ACM, 2019.

[26] A. Verma, M. Korupolu, and J. Wilkes, "Evaluating job packing in warehouse-scale computing," in *2014 IEEE International Conference on Cluster Computing (CLUSTER 2014), Madrid, Spain, September 22-26, 2014*, pp. 48–56, IEEE Computer Society, 2014.

[27] NVIDIA, "Nvidia titan rtx," 2019. https://www.nvidia.com/content/dam/en-zz/Solutions/titan/documents/titan-rtx-for-creators-us-nvidia-1011126-r6-web.pdf, Accessed: 23.04.2024.

[28] NVIDIA, "Geforce gtx 1080 ti," 2021. https://www.nvidia.com/en-gb/geforce/graphics-cards/geforce-gtx-1080-ti/specifications/, Accessed: 23.04.2024.

[29] Dell, "Dell emc poweredge t640," 2020. https://i.dell.com/sites/csdocuments/Product_Docs/en/poweredge-t640-technical-guide.pdf, Accessed: 23.04.2024.

[30] S. Ilager, K. Ramamohanarao, and R. Buyya, "Thermal prediction for efficient energy management of clouds using machine learning," *IEEE Transactions on Parallel and Distributed Systems (TPDS 2021)*, vol. 32, no. 5, pp. 1044–1056, 2021.

[31] A. Valadarsky, G. Shahaf, M. Dinitz, and M. Schapira, "Xpander: Towards optimal-performance datacenters," in *Proceedings of the 12th International on Conference on emerging Networking EXperiments and Technologies (CoNEXT 2016), Irvine, California, USA, December 12-15, 2016*, pp. 205–219, ACM, 2016.

[32] F. Mastenbroek, G. Andreadis, S. Jounaid, W. Lai, J. Burley, J. Bosch, E. V. Eyk, L. Versluis, V. van Beek, and A. Iosup, "Opendc 2.0: Convenient modeling and simulation of emerging technologies in cloud datacenters," in *21st IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGrid 2021), Melbourne, Australia, May 10-13, 2021*, pp. 455–464, IEEE, 2021.

[33] X. Chu, S. Talluri, L. Versluis, and A. Iosup, "How do ML jobs fail in datacenters? Analysis of a long-term dataset from an HPC cluster," in *Companion of the 2023 ACM/SPEC International Conference on Performance Engineering (ICPE 2023), Coimbra, Portugal, April 15-19, 2023*, pp. 263–268, ACM, 2023.

[34] R. Garg, T. Patel, G. Cooperman, and D. Tiwari, "Shiraz: Exploiting system reliability and application resilience characteristics to improve large scale system throughput," in *48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2018), Luxembourg City, Luxembourg, June 25-28, 2018*, pp. 83–94, IEEE Computer Society, 2018.

[35] N. Yigitbasi, M. Gallet, D. Kondo, A. Iosup, and D. Epema, "Analysis and modeling of time-correlated failures in large-scale distributed systems," in *2010 11th IEEE/ACM International Conference on Grid Computing (CCGRID 2010)*, pp. 65–72, IEEE, 2010.

[36] A. K. Paul, A. M. Karimi, and F. Wang, "Characterizing machine learning I/O workloads on leadership scale HPC systems," in *29th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS 2021), Houston, TX, USA, November 3-5, 2021*, pp. 1–8, IEEE, 2021.

[37] E. Cortez, A. Bonde, A. Muzio, M. Russinovich, M. Fontoura, and R. Bianchini, "Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms," in *Proceedings of the 26th Symposium on Operating Systems Principles (SOSP 2017), Shanghai, China, October 28-31, 2017*, pp. 153–167, ACM, 2017.

[38] G. P. Rodrigo, P. Östberg, E. Elmroth, K. Antypas, R. A. Gerber, and L. Ramakrishnan, "Towards understanding HPC users and systems: A NERSC case study (JPDC 2018)," *Journal of Parallel and Distributed Computing*, vol. 111, pp. 206–221, 2018.

[39] Z. Liu, R. Lewis, R. Kettimuthu, K. Harms, P. H. Carns, N. S. V. Rao, I. T. Foster, and M. E. Papka, "Characterization and identification of HPC applications at leadership computing facility," in *2020 International Conference on Supercomputing (ICS 2020), Barcelona Spain, June, 2020*, pp. 29:1–29:12, ACM, 2020.