[Alex@encoded.vc](mailto:Alex@encoded.vc)

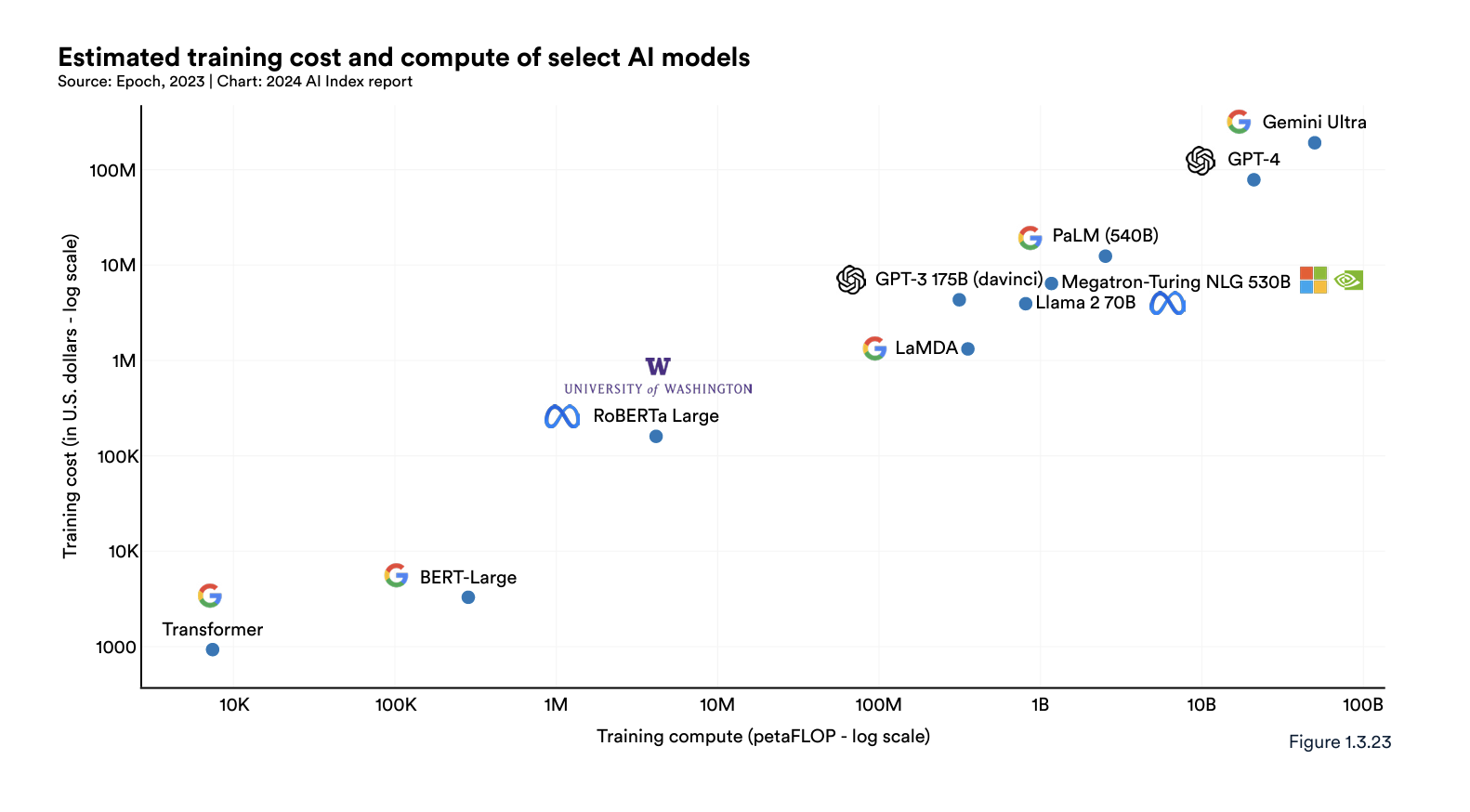
Work in progress

LLM infra utilization, reliability, and observability

**Where my FLOPs at?**

Everyone knows that 10s-100s of billions of dollars are being spent and planned on data centers and infrastructure for training large LLMs. What is less obvious to those not deep in the details of these infrastructures is how inefficiently these FLOPS, watts, and dollars are being utilized.

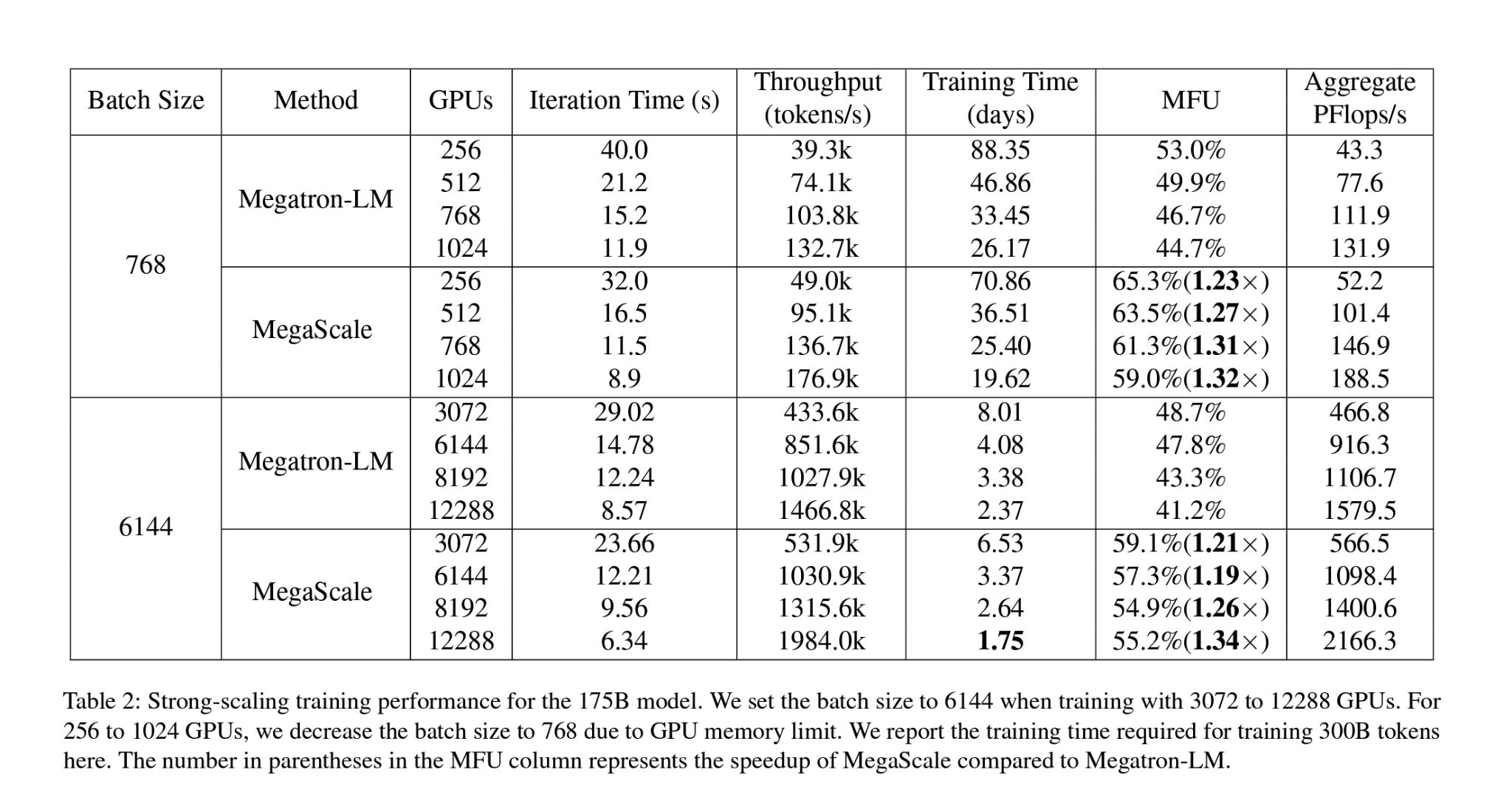
The characteristics of the training workload are much different than large scale SAAS or even search backends and much closer to traditional HPC applications. SaaS backends are loosely coupled and typically composed of a set of microservices communicating asynchronously over a set of RPC protocols. Training is a tightly coupled application, operating across many machines with a network interconnect. Each GPU needs to do it’s computation and exchange weights with the rest of the cluster over the network. Progress happens synchronously, compute → update state of the model. . As the size of the cluster increases, failures become more common. This leads to the need to frequently checkpoint the state of the model and save it to storage to reload in the case of failures. There are lessons to be learned from the traditional HPC/Supercomputing community who have been deals with these types of problems, at large scale, and using GPUs, for many years.



While data on hardware utilization is increasingly hard to come by in the ClosedAI era, this [chart](https://docs.google.com/spreadsheets/d/1uFBmqPC8IJKOi0Dmky_qBU6A3pJDltIg4MIANV21wNI/edit?usp=sharing) below compiles public data from the [Epoch database](https://epochai.org/data/epochdb/visualization). It’s not surprising that Google operates the most efficient infrastructure as they have long known the [datacenter is the computer.](https://research.google/pubs/the-datacenter-as-a-computer-an-introduction-to-the-design-of-warehouse-scale-machines-second-edition/) But serious organizations spending billions on hardware and talent are routinely in the 30-40%. This public data jibes directionally with conversations I’ve had with practitioners training models at MosaicML/Databricks, Suno, and others. They are working hard to be in the 40%, especially on public clouds where they have less control over the infrastructure, even when using bare metal.

| **System** | **Organization** | **Training compute (FLOP)** | **Hardware utilization** | **Parameters** | **Training hardware** |
| --- | --- | --- | --- | --- | --- |
| LaMDA | Google | 3.55E+23 | 56.50% | 137,000,000,000 | Google TPU v3 |
| AlexaTM 20B | Amazon | 2.04E+23 | 49.35% | 19,750,000,000 | NVIDIA A100 |
| BLOOM-176B | Hugging Face,BigScience | 5.77E+23 | 48.08% | 176,247,271,424 | NVIDIA A100 SXM4 80 GB |
| LLaMA-65B | Meta AI | 5.50E+23 | 47.46% | 65,200,000,000 | NVIDIA A100 |
| OPT-175B | Meta AI | 4.30E+23 | 47.12% | 175,000,000,000 | NVIDIA A100 SXM4 80 GB |
| PaLM (540B) | Google Research | 2.53E+24 | 46.20% | 540,350,000,000 | Google TPU v4 |
| Skywork-13B | Kunlun Inc. | 2.50E+23 | 46.00% | 13,000,000,000 | NVIDIA A800 |
| Yuan 1.0 | Inspur | 3.54E+23 | 45.00% | 245,730,000,000 | NA |
| Code Llama-70B | Meta AI | 1.23E+24 | 43.50% | 70,000,000,000 | NVIDIA A100 SXM4 80 GB |
| Llama 2-70B | Meta AI | 8.10E+23 | 43.50% | 70,000,000,000 | NVIDIA A100 SXM4 80 GB |
| GLM-130B | Tsinghua University | 3.78E+23 | 43.30% | 130,000,000,000 | NVIDIA A100 SXM4 40 GB |
| Falcon-40B | Technology Innovation Institute | 2.40E+23 | 38.64% | 40,000,000,000 | NVIDIA A100 |
| Gopher (280B) | DeepMind | 6.31E+23 | 37.80% | 280,000,000,000 | Google TPU v3 |
| Meena | Google Brain | 1.12E+23 | 34.39% | 2,600,000,000 | Google TPU v3 |
| Nemotron-3-8B | NVIDIA | 1.80E+23 | 34.00% | 8,000,000,000 | NVIDIA A100 |
| BloombergGPT | Bloomberg,Johns Hopkins University | 2.36E+23 | 32.00% | 50,558,868,480 | NVIDIA A100 |
| UL2 | Google Research,Google Brain | 1.20E+23 | 31.80% | 20,000,000,000 | Google TPU v4 |
| Megatron-Turing NLG 530B | Microsoft,NVIDIA | 1.17E+24 | 30.20% | 530,000,000,000 | NVIDIA A100 SXM4 80 GB |
| GPT-3 175B (davinci) | OpenAI | 3.14E+23 | 21.96% | 175,000,000,000 | NVIDIA Tesla V100 DGXS 32 GB |
| HyperCLOVA | NAVER,Search Solutions | 1.48E+23 | 20.00% | 82,000,000,000 | NVIDIA A100 |
| Falcon-180B | Technology Innovation Institute | 3.76E+24 | 18.76% | 180,000,000,000 | NVIDIA A100 SXM4 40 GB |

These figures from a recent Bytedance paper, [MegaScale](https://arxiv.org/html/2402.15627v1), are very impressive.



**So, where my FLOPS at?**

Like many systems optimization problems, the answer isn’t obvious and bottlenecks appear up and down the stack:

* Memory bandwidth between the GPU and HBM
* Size of the NVLink / NVSwitch domain. [UALink by Intel/AMD/BRCM](https://www.phoronix.com/news/Ultra-Accelerator-Link-UALink)
* Optimization of collective operations like [NCCL](https://developer.nvidia.com/nccl), [MCCL](https://github.com/microsoft/msccl), etc
* Number of checkpoint restarts due to hardware failures and time to write and reload from storage
* Scheduling and placement of training jobs - SLURM, K8s, custom schedulers, etc
* Network bandwidth and congestion for models spread across top-of-rack and spine switches - See Google optical switching and optimized congestion control protocols.
* Choice of parallelism techniques
* Others?

SNOW recently published a detailed blog on performance issues they need to overcome in training Artic. The section on communications overhead was particularly striking, [“Without any further optimizations, we observed the communication overhead to be larger than 50%.”](https://medium.com/snowflake/snowflake-arctic-cookbook-series-building-an-efficient-training-system-for-arctic-6658b9bdfcae)

**Reliability and Failures**

The other issue eating FLOPs is reliability. The field failure rate of hardware and cluster instability are all leaving a significant amount of FLOPs on the floor. The quotes below are consistent with practitioners I’ve been speaking with regarding field failure rates and overall cluster instability.

A [recent blog by Reka](https://www.yitay.net/blog/training-great-llms-entirely-from-ground-zero-in-the-wilderness) co-founder and chief scientist Yi Tay paints a startling picture, Tay points out that training clusters are literally a, “Hardware Lottery,” making a reference to the [fascinating paper of the same name](https://arxiv.org/pdf/2009.06489) by his former colleague at Google Brain, Sara Hooker.

*“More specifically, we’ve leased a few clusters from several compute providers, each with a range of hundreds to thousands of chips. We’ve seen clusters that range from passable (just annoying problems that are solvable with some minor SWE hours) to totally unusable clusters that fail every few hours due to a myriad of reasons. Specifically, some clusters have nodes that fail every N hour with issues ranging from cabling issues (where N is unreasonably small), GPU hardware errors etc. Even more surprisingly, every cluster across the same provider could also be vastly different in terms of how robust it was. “*

Some perspective from X.ai, Meta, and BigScience:

**X.ai - Grok**

*“LLM training runs like a freight train thundering ahead; if one car derails, the entire train is dragged off the tracks, making it difficult to set upright again. There are a myriad of ways GPUs fail: manufacturing defects, loose connections, incorrect configuration, degraded memory chips, the occasional random bit flip, and more. When training, we synchronize computations across tens of thousands of GPUs for months on end, and all these failure modes become frequent due to scale. To overcome these challenges, we employ a set of custom distributed systems that ensure that every type of failure is immediately identified and automatically handled. At xAI, we have made maximizing useful compute per watt the key focus of our efforts. Over the past few months, our infrastructure has enabled us to minimize downtime and maintain a high Model Flop Utilization (MFU) even in the presence of unreliable hardware.”*

[*https://x.ai/blog/grok*](https://x.ai/blog/grok)

**Meta OPT-175B trained on 992 80GB A100s**

“We faced a significant number of hardware failures in our compute cluster while training OPT-175B. In total, hardware failures contributed to at least 35 manual restarts and the cycling of over 100 hosts over the course of 2 months. During manual restarts, the training run was paused, and a series of diagnostics tests were conducted to detect problematic nodes. Flagged nodes were then cordoned off and training was resumed from the last saved checkpoint. Given the difference between the number of hosts cycled out and the number of manual restarts, we estimate 70+ automatic restarts due to hardware failures.”

<https://arxiv.org/abs/2205.01068> and the training [Logbook](https://github.com/facebookresearch/metaseq/blob/main/projects/OPT/chronicles/OPT175B_Logbook.pdf)

**BigScience - Bloom**

*“Every week or so we have hardware issues where one of the GPUs dies. Most of the time the SLURM job auto-restarts and we lose at most 3 hours of training, since we save a checkpoint every 100 iterations, which takes about 3 hours to train.”*

*“We can plan to use 384 GPUs out of 416 as 4 nodes of 8 GPUs need to remain reserved for when some nodes happen to be down.”*

<https://github.com/bigscience-workshop/bigscience/blob/master/train/tr11-176B-ml/chronicles.md>

These stories map to conversations I’ve had with people in the guts of training large models:

“I’ve seen 10-15 % of nodes have failures, 4k GPU cluster maxed out at 3.2k active GPUs.

400 out of 500 in a small cluster. “ MosaicML

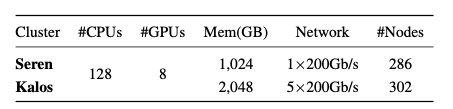
“When hardware fails, the telemetry you have is garbage. In our 512 GPU cluster on Oracle, we

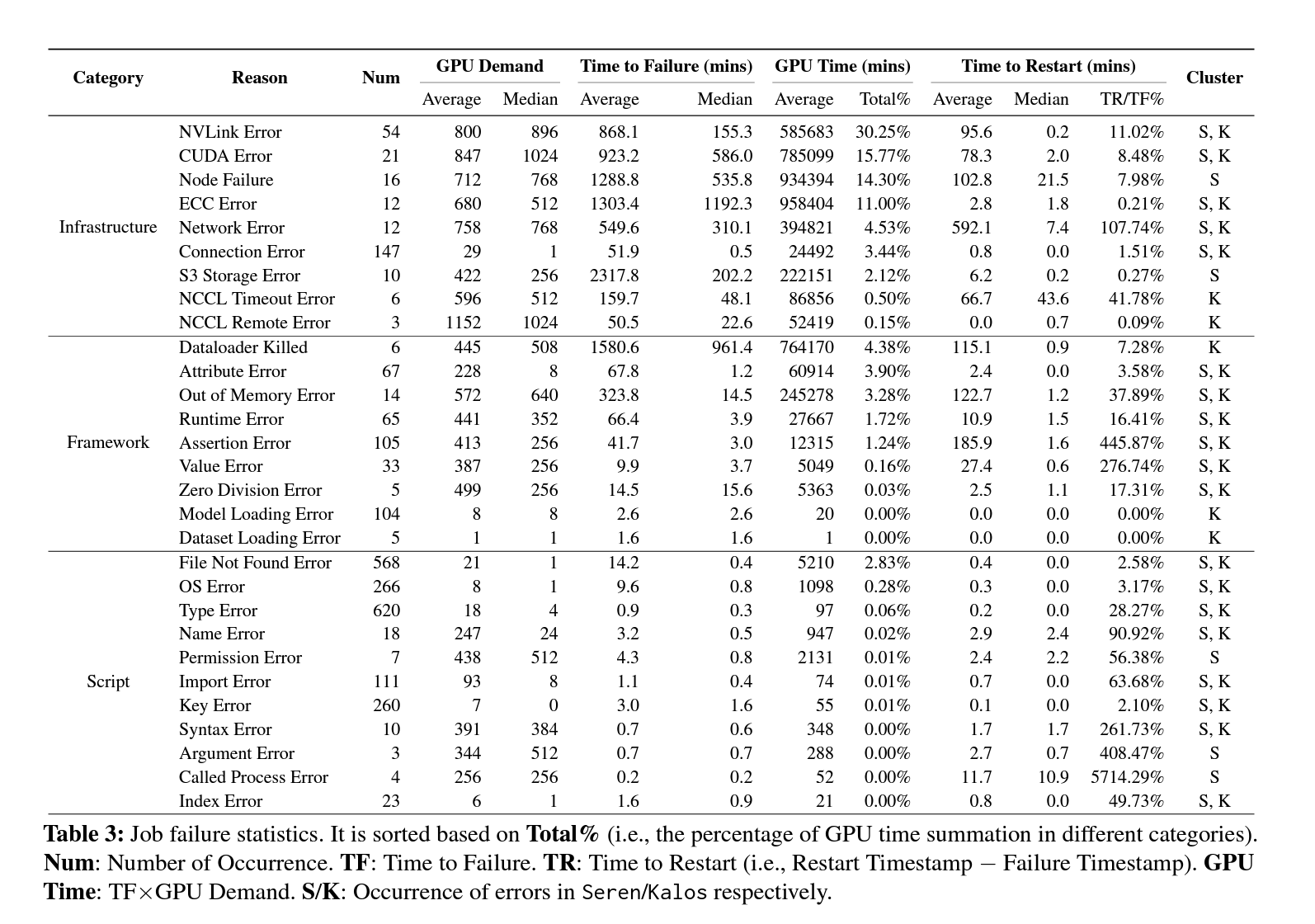
typically leave 24 idle because training jobs will die and we need to restart them

can't start again with fewer nodes. better to eat the cost of a few spare nodes or have them run low priority jobs” - Audio AI start-up

[This Blog from Imbue](https://imbue.com/research/70b-infrastructure/) on their operational experience training a 70B model is an absolute goldmine.

[Characterization of Large Language Model Development in the Datacente](https://www.usenix.org/conference/nsdi24/presentation/hu)r from Shanghai AI Laboratory, NSDI 2024, presents a fascinating breakdown of the cause of job failures in two reasonable size training clusters, Seren and Kalos.





**Observability**

It’s also clear that there is a lack of visibility into cluster behavior across the layers. The tightly coupled nature of these workloads compared to SaaS apps means small perturbations and have a serious impact on run-time and $.

NVDA does have some profiling and observability tools [DCGM](https://github.com/NVIDIA/DCGM), [Nsight](https://developer.nvidia.com/nsight-systems), [BCM](https://docs.nvidia.com/base-command-manager/index.html#product-manuals) (via acquisition of Bright Computing) but they don’t seem to be broadly adopted. It doesn’t appeal to be a cohesive solution. You can pipe DCGM into [DDOG](https://www.datadoghq.com/monitoring/nvidia-gpu-monitoring/) but this is just dashboards from NVDA chip level metrics. Arista has also recognized the observability gap and announced [new tools trying](https://blogs.arista.com/blog/ai-center) to bridge host and networking views. The latest product has an EOS based agent running on MLNX DPUs but other details are limited.

<https://developer.nvidia.com/management-library-nvml>

It feels like there could be an opportunity to build an observability tool that bridges telemetry from across the infrastructure layers, GPU/CPU, server, storage, network libraries (NCCL) and physical network scheduling, informed by model architecture. Cross section your cluster, introspect it at everylayer. Unclear if you can get enough of the data required from managed cloud GPU services or if the market would be limited to those running their own infra or using bare metal clouds.