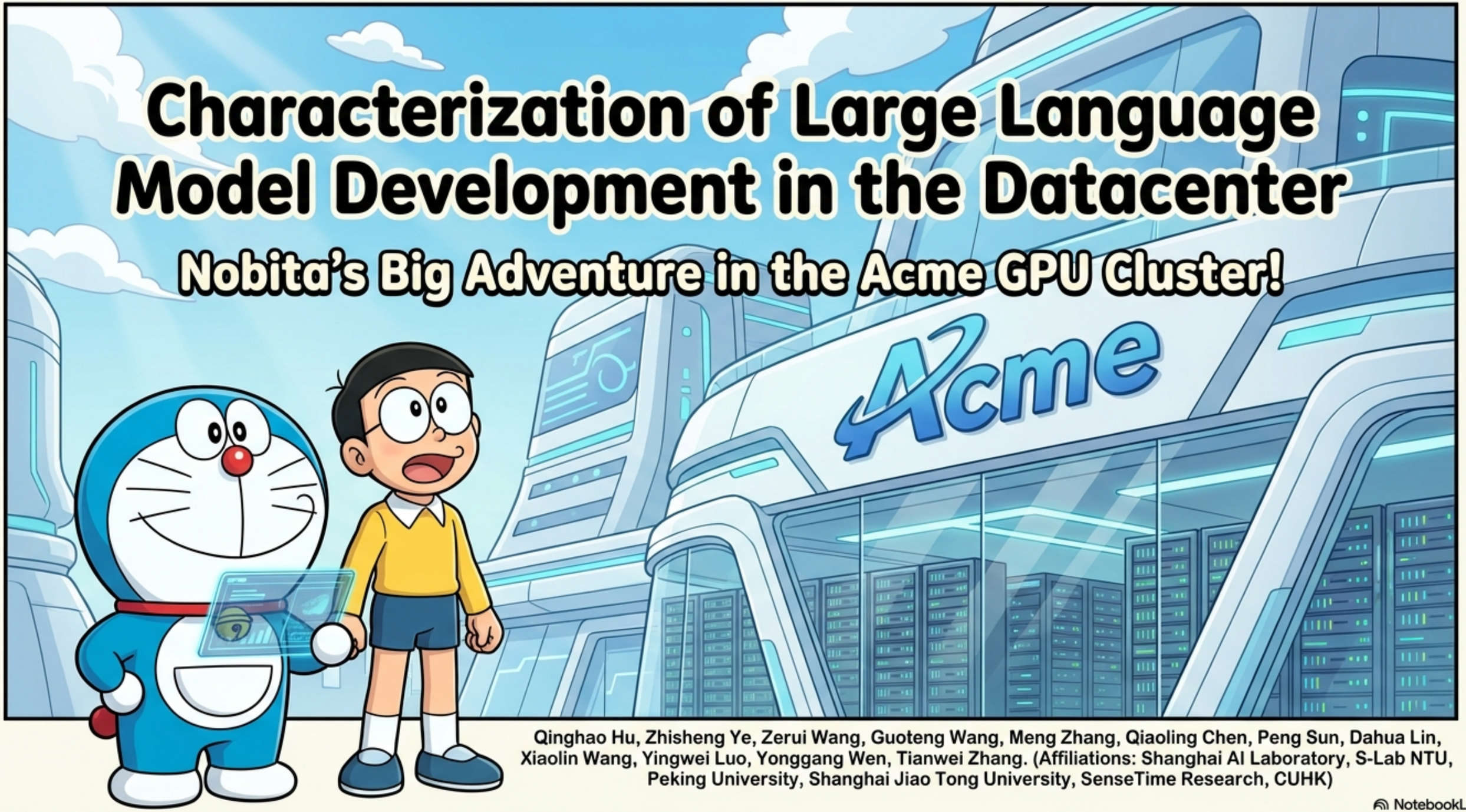
Characterization of Large Language Model Development in the Datacenter

Nobita's Big Adventure in the Acme GPU Cluster!



Qinghao Hu, Zhisheng Ye, Zerui Wang, Guoteng Wang, Meng Zhang, Qiaoling Chen, Peng Sun, Dahua Lin, Xiaolin Wang, Yingwei Luo, Yonggang Wen, Tianwei Zhang. (Affiliations: Shanghai AI Laboratory, S-Lab NTU, Peking University, Shanghai Jiao Tong University, SenseTime Research, CUHK)

Doraemon's Plan!

1. **The LLM Dream:**
   The World of LLM Development

2. **A Look Inside Acme:**
   Datacenter Characterization

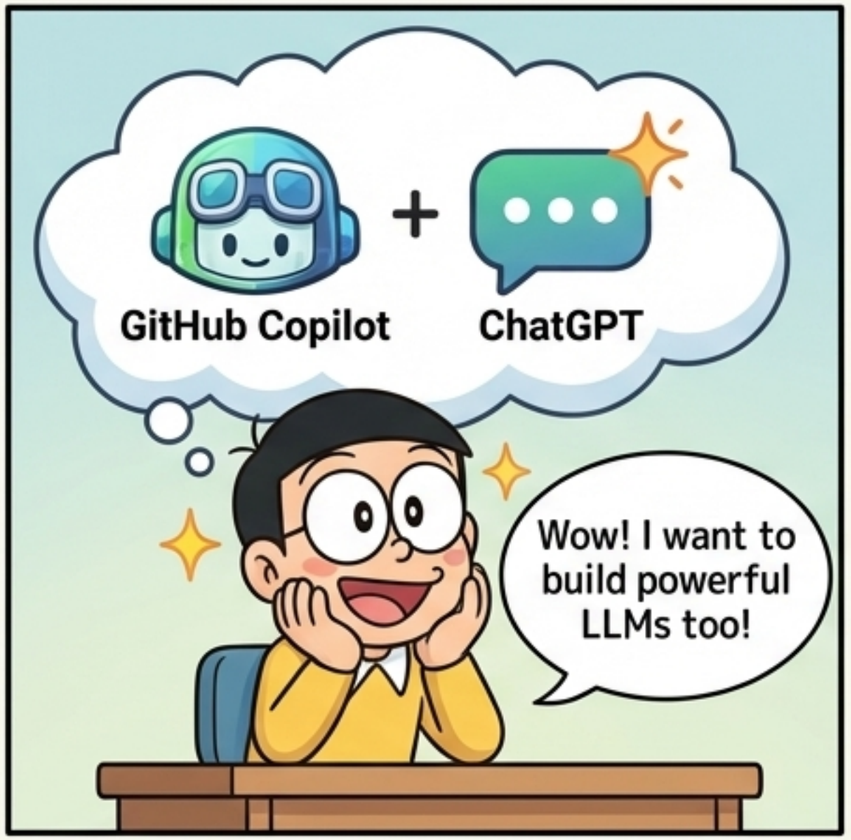
3. **The Main Characters:**
   Profiling Pretraining & Evaluation

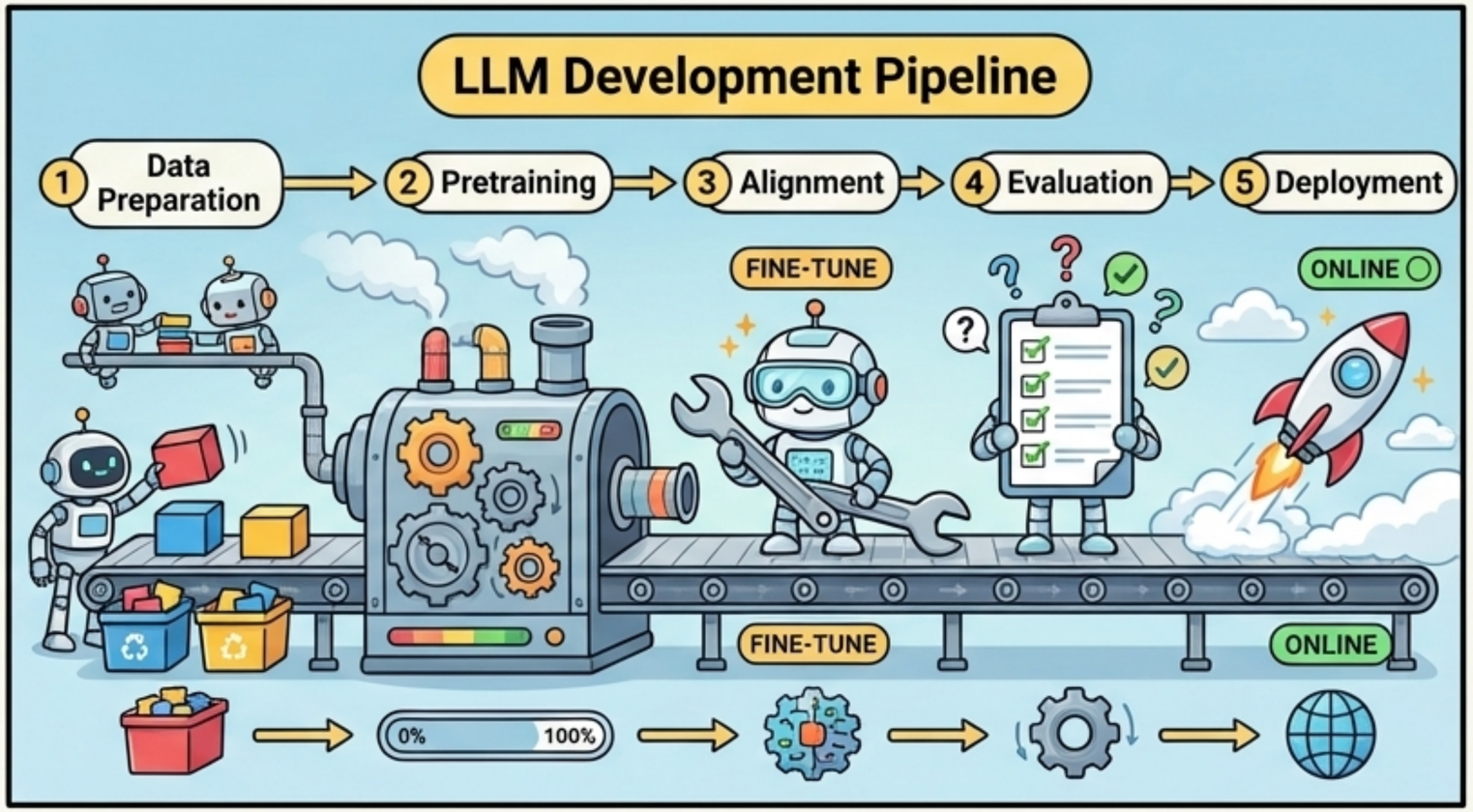
4. **The Trouble Makers:**
   A Deep Dive into Failures

5. **Doraemon's Gadgets!**
   Our Deployed Systems

6. **The Future is Bright!**
   Conclusion & Takeaways

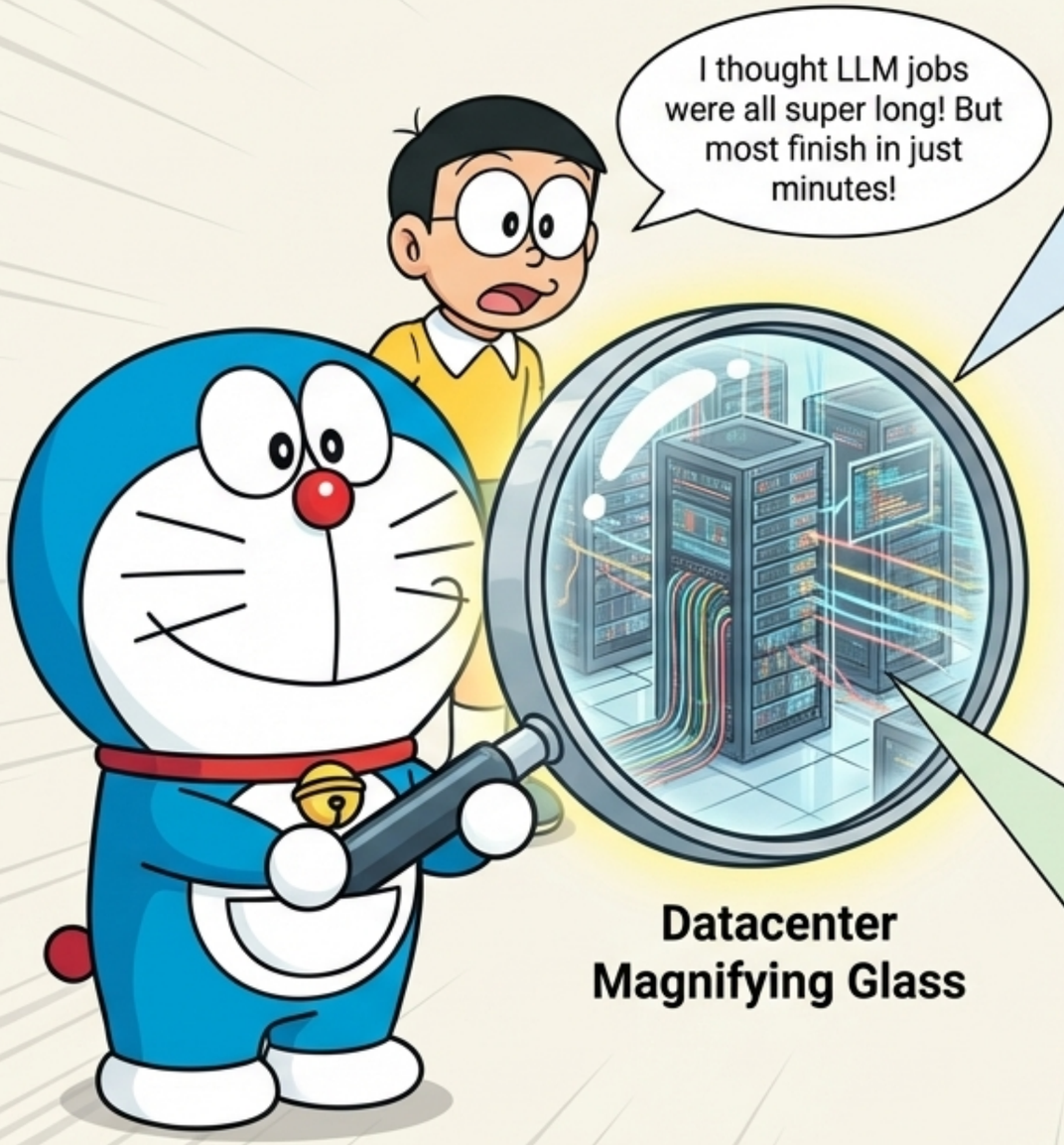
The Grand Challenge of Building LLMs





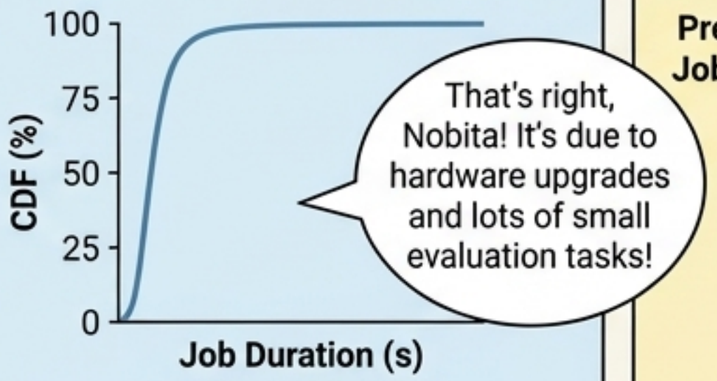
Developing Large Language Models is a complex, multi-stage pipeline. From preparing massive datasets to intensive pretraining and evaluation cycles, it requires a huge and expensive GPU infrastructure. This journey is often riddled with numerous challenges like frequent hardware failures, intricate parallelization strategies, and imbalanced resource utilization.

What's *Really* Happening Inside the Datacenter?



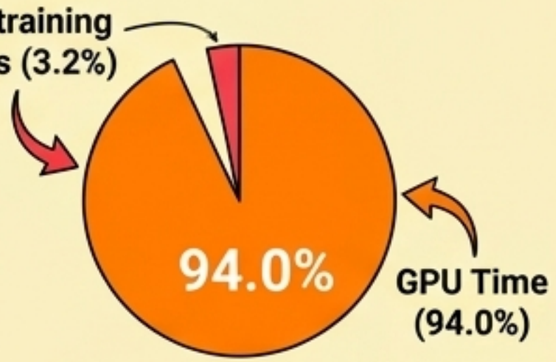
Datacenter Magnifying Glass

Shorter Job Duration



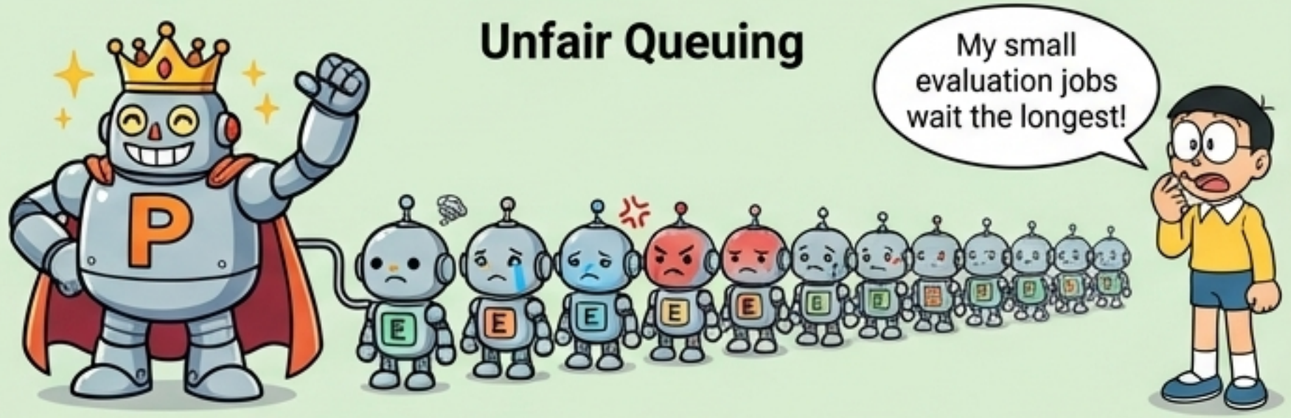
Median job duration is just 2 minutes, 2.7–12.8x shorter than prior DL workloads.

Imbalanced Workload



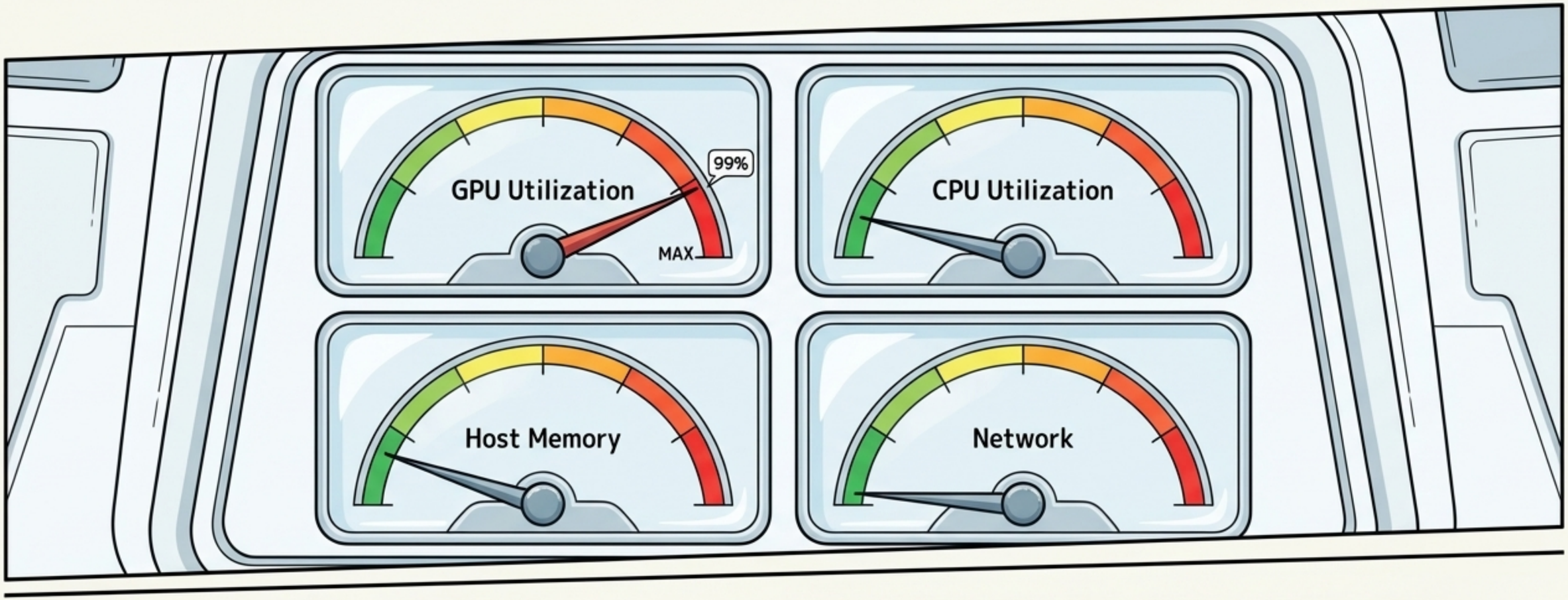
A few giant pretraining jobs consume almost all the resources in our Kalos cluster!

Unfair Queuing



Resources are reserved for big jobs, so small but crucial evaluation tasks face the longest queuing delays.

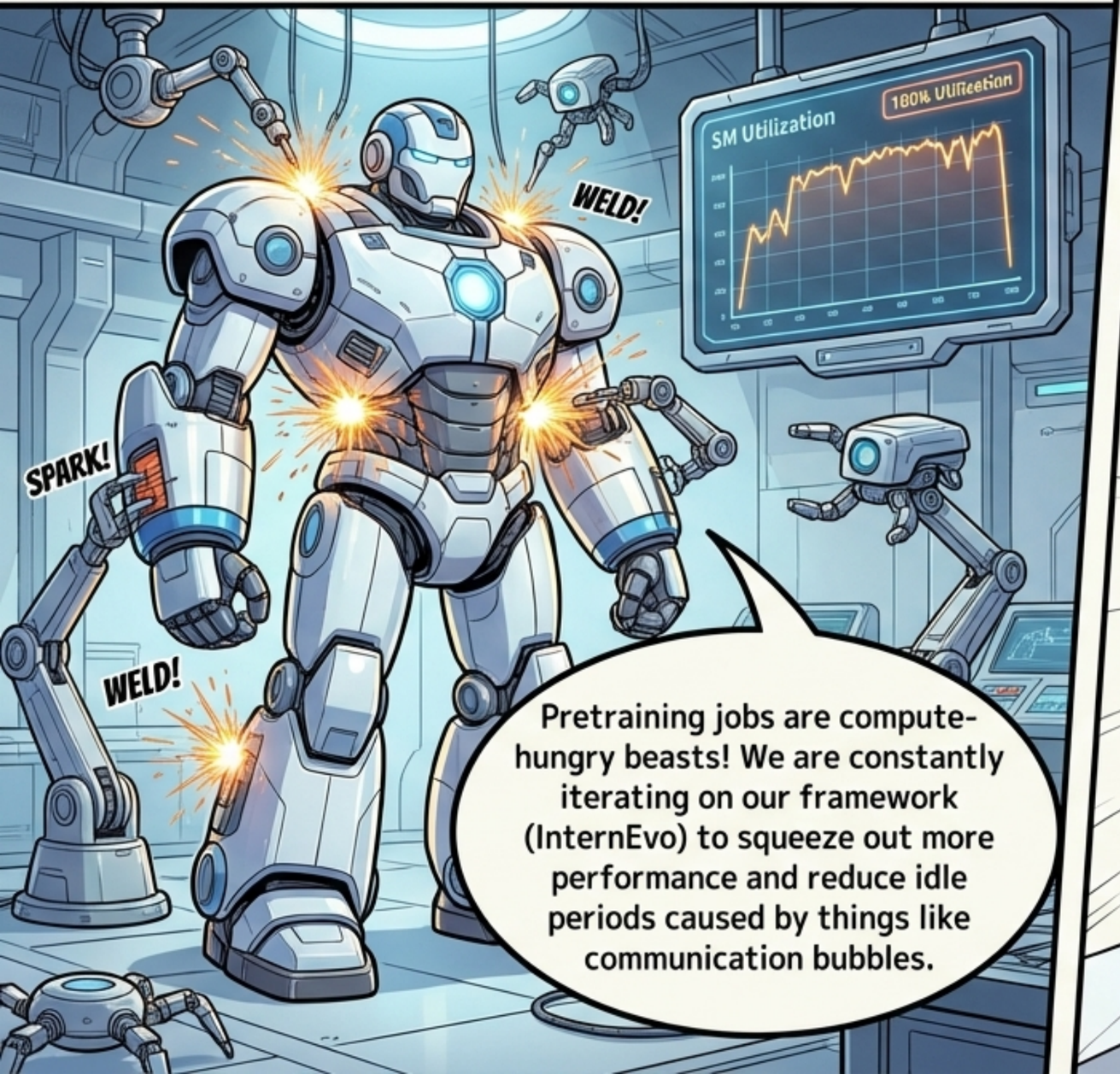
The GPU is Working Hard... But Its Friends are Slacking!



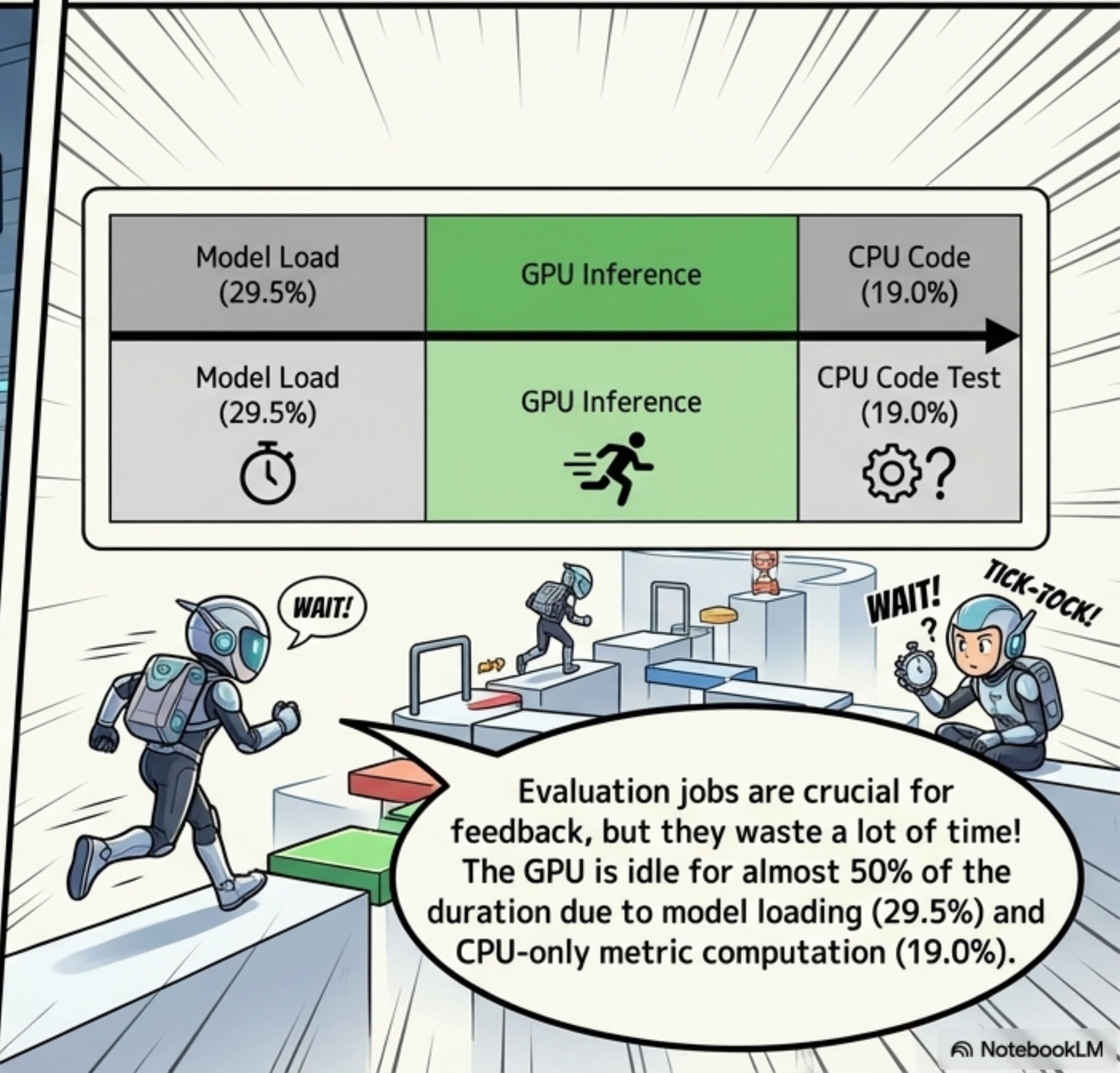
Our investigation shows that while GPUs are running at full throttle (Median GPU utilization of 99% in Kalos), other associated resources like CPU, host memory, and network are frequently underutilized.

This confirms LLMs are computationally and memory-intensive on the GPU. But it also reveals an opportunity: can we use that idle CPU and Memory for something useful?

Pretraining: The Heavy Lifter



Evaluation: The Speedy Messenger



Attack of the Failure Monsters!

WANTED: INFRA-BEAST

Accounts for over 82% of lost GPU time! The most dangerous villain!

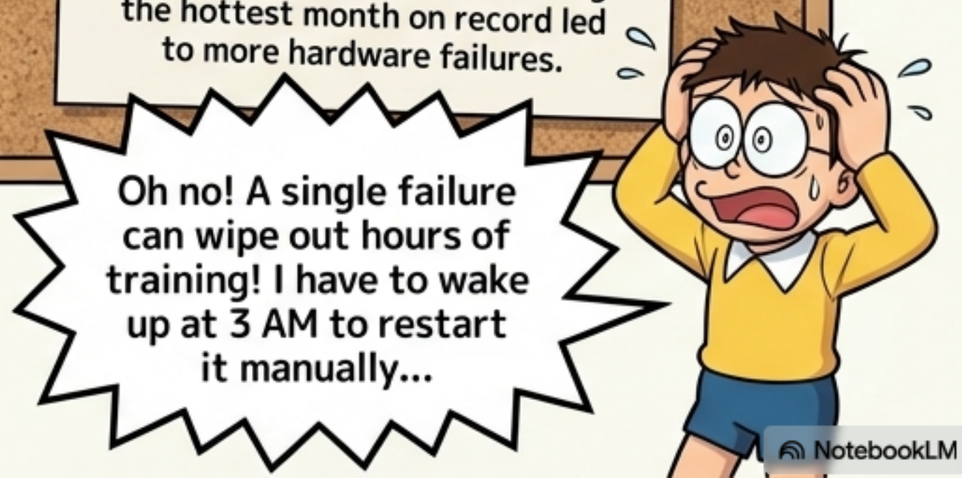
WANTED: CODE GREMLIN

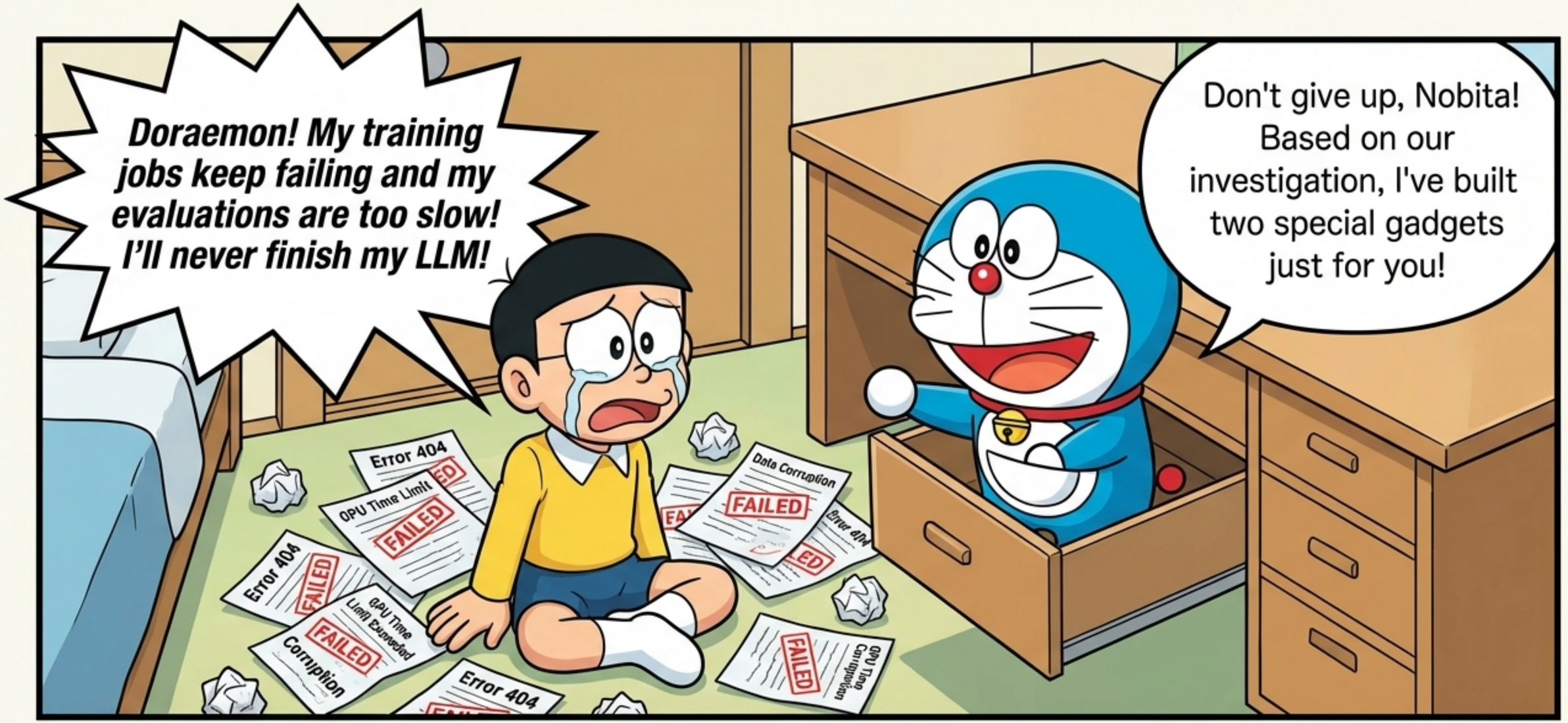
The most common type of failure, but usually quick to fix.

WANTED: THE HEAT WAVE

A surprising foe! A 5°C rise in server room temperature during the hottest month on record led to more hardware failures.

~40% of all jobs FAIL!

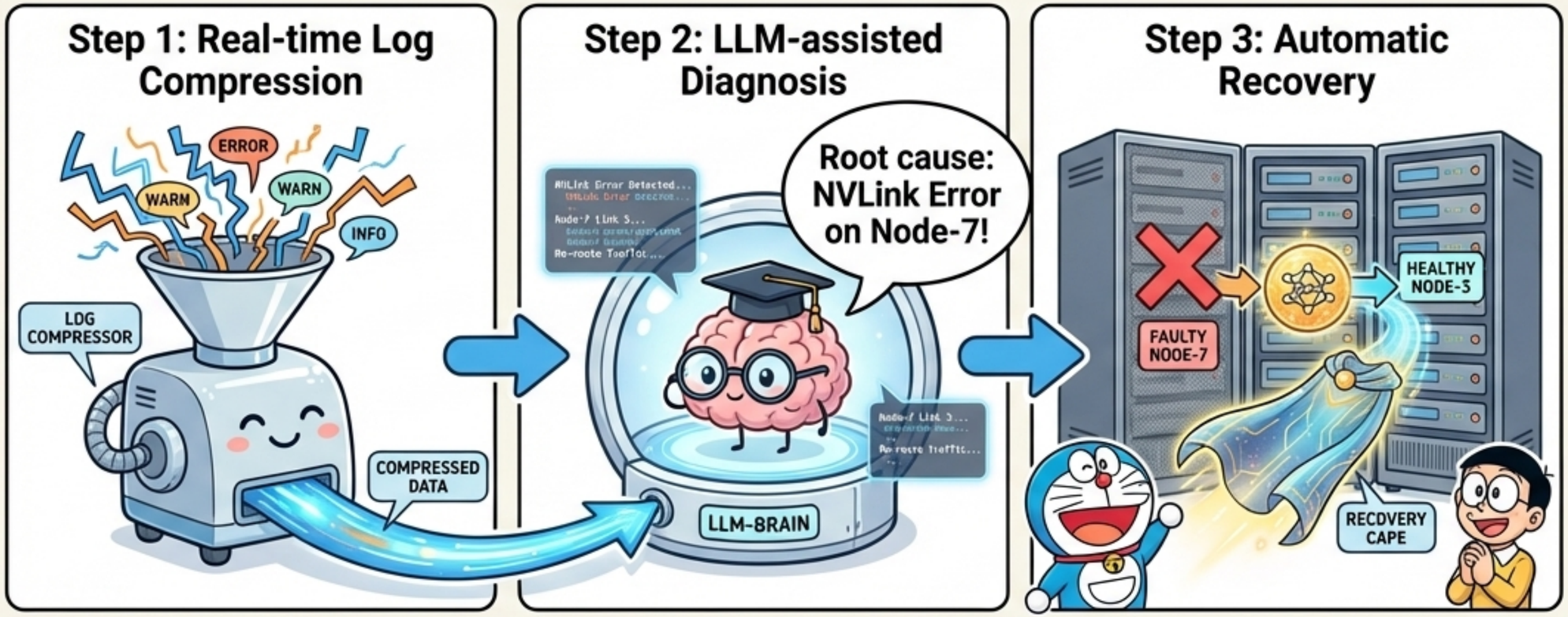




Our deep characterization revealed clear opportunities for improvement. We built two systems, systems, integrated into our LLM framework, to make development more robust and efficient.

Gadget #1: The Auto-Recovery Cape (Fault-tolerant Pretraining)

Function: Automatically diagnoses failures and recovers training jobs.



Step 1: Real-time Log Compression

Step 2: LLM-assisted Diagnosis

Step 3: Automatic Recovery

Key Features

"Asynchronous Checkpointing": Saves progress frequently without blocking training (accelerates checkpointing by 3.6x to 58.7x!).

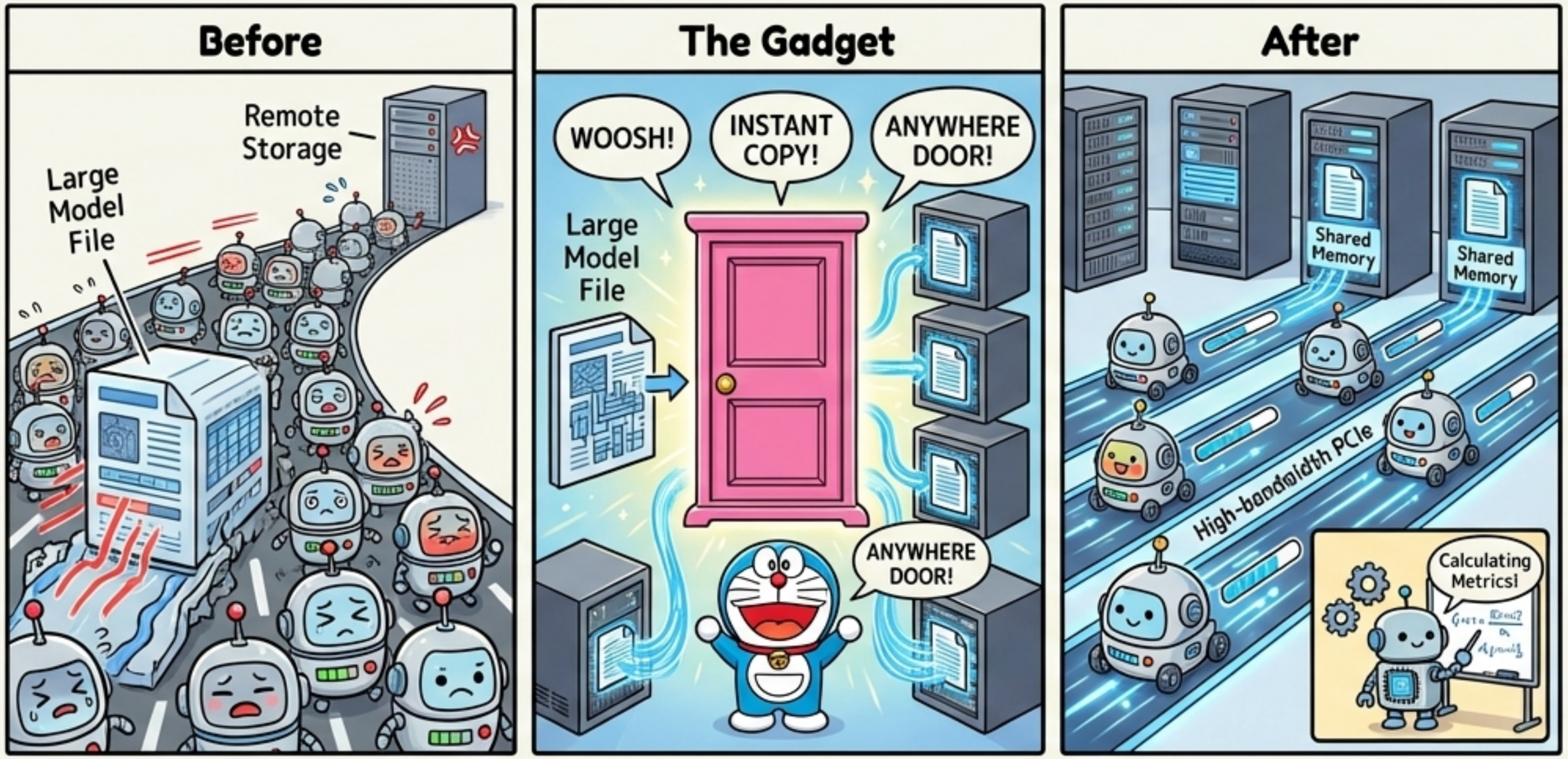
"LLM-Powered Diagnosis": Uses an LLM to accurately understand complex error logs and identify root causes.

"Automatic Recovery": Pinpoints faulty hardware and restarts the job from the last checkpoint, no human needed!

Reduces manual intervention by ~90%!
No more 3 AM wake-up calls!

Gadget #2: The Anywhere Door (Decoupled Scheduling for Evaluation)

Function: Speeds up evaluation to provide fast feedback on model quality.



Key Features

⏱️ **Decoupled Model Loading:**

- Loads the model *once* to each node's shared memory, avoiding remote storage contention and network bottlenecks.

🧠 **Decoupled Metric Computation:**

- Offloads CPU-only work to separate jobs, minimizing GPU idle time.

⚖️ **Prior-based Elastic Scheduling:**

- Balances the workload across all GPUs for maximum efficiency.

Reduces total evaluation makespan by up to 1.8x!
Get feedback on your model in a flash! 🔥

A New Era for LLM Development!

Summary of Findings:

LLM workloads are unique: Characterized by many **short jobs, extreme resource skew** (a few pretraining jobs use 94% of GPU time), and **frequent failures** (~40% of jobs fail).

A deep, **data-driven understanding** is the key to **identifying the real bottlenecks and opportunities** for optimization.

Summary of Contributions:

Our **Fault-Tolerant System** automates failure diagnosis and recovery, reducing manual intervention by ~90%.

Our **Decoupled Evaluation Scheduler** accelerates feedback by up to 1.8x, resolving I/O bottlenecks and minimizing GPU idle time.

"By understanding the unique character of LLM workloads, we can build smarter systems to accelerate research and development for everyone."

Want to Build Your Own Gadgets?

We believe our lessons and insights can benefit the entire community. Our traces, systems, and models are publicly available to help acolerate LLM research everywhere!

The AcmeTrace Dataset

https://github.com/InternLM/AcmeTrace

Our System (InternEvo)

https://github.com/InternLM/InternEvo

Our Models (InternLM)

https://huggingface.co/internlm