Observability and Fault Tolerance for LLM training

Odej Kao

Distributed and Operating Systems
Technische Universität Berlin

Mitigating Training Instabilities

- LLMs as core technology with expensive (1000xGPUs, huge data sets) and long-lasting (weeks, months) training to produce models with hundreds of billions parameters

- Significant impact of training instabilities due to
  - Increased likelihood of hardware faults, impacting training time and cost
  - Loss fluctuations associated with slow convergence / non-convergence (loss spikes)

Component	Interruption Count	% of Interruptions
Faulty GPU	148	30.1%
GPU HBM3 Memory	72	17.2%
Software Bug	54	12.9%
Network Switch/Cable	35	8.4%
Host Maintenance	32	7.6%
GPU SRAM Memory	19	4.5%
GPU System Processor	17	4.1%
NIC	7	1.7%
NCCL Watchdog Timeouts	7	1.7%
Silent Data Corruption	6	1.4%
GPU Thermal Interface + Sensor	6	1.4%
SSD	3	0.7%
Power Supply	3	0.7%
Server Chassis	2	0.5%
IO Expansion Board	2	0.5%
Dependency	2	0.5%
CPU	2	0.5%
System Memory	2	0.5%

Silent Data Corruptions

- Hardware faults relate to CPU/GPU, communication, memory
  - ➢ Error signals available → challenging root cause detection
  - ➢ Hardware failing without sending error signals → Silent data corruption

⇒ SDCs can lead models to converge to different optima with different weights and even cause spikes in the training loss

- Research goal

Develop methods to identify and localize SDCs in the presence of LLM training instabilities, particularly loss spikes, and provide non-preemptive mitigation strategies

Research Questions

- RQ1: Which types of silent data corruptions can cause training/inferencing instabilities?
- RQ2: How to discover relevant silent data corruptions?
- RQ3: Which elastic techniques allow to mitigate interruption and continue training?

- Midterm vision: Predicting and detecting silent data corruptions and corresponding training instabilities.

First Steps

- In-depth study
  - Metrics (e.g. gradient norm, query and key vectors, entropy attention matrix, cosine similarity, …)
  - Mitigation strategies (e.g. parametric singularity smoothing, QK normalization, decreased learning rate, gradient clipping, …)

- Setting-up research environment

- Challenges
  - Access to data
  - Developing fault injectors

Thank you!