CS 181AI
Lecture 23

Profiling + Debugging

Arthi Padmanabhan

Apr 12, 2023

Last Time

- Strategies for Gradient Aggregation in Synchronous Distributed Training
    - Ring All-Reduce
    - GCP Server

Today

- Asynchronous Distributed Training (finish)
- Consistency Models
- Using a profiler

Asynchronous Gradient Update

- Keep one server for all gradients -> parameter server model
- Each worker computes gradients and makes updates asynchronously (no waiting for each other)

Parameter Server

Asynchronous Gradient Update

- Keep one server for all gradients -> parameter server model
- Each worker computes gradients and makes updates asynchronously (no waiting for each other)
- What could go wrong?

Asynchronous Gradient Update

- Keep one server for all gradients -> parameter server model
- Each worker computes gradients and makes updates asynchronously (no waiting for each other)
- Workers could be taking a stale version of parameters and computing gradients on those!

Consistency Models

- Strong
- Weak
- Eventual
- Bounded

Profiling

- “The real problem is that programmers have spent far too much time worrying about efficiency in the wrong places at the wrong times”

Why Profiling

- It’s difficult to figure out why a program is slow
- Reading the code can be misleading
- CPUs and GPUs are complex
- Measure!

What can profiling do?

- Identify hotspots
- Visualize where time is being spent
- Collect metrics

Sampling

- Every so often while program is running, profiler interrupts to get the stack trace

- Profiling results are based on statistical averages

- Minimal overhead

Instrumentation

- Add code hooks to explicitly record metrics for certain portion of code
- Profiling results are actual data, not averages
- Little harder to use

PyTorch Profiler

- Built into PyTorch
- Open Profiling.ipynb in Lec 23 resources
- We need to specify several things before running:
  - Wait
  - Warmup
  - Active
  - Repeat

Acknowledgments

- Nikita Namjoshi, Google Cloud Developer Advocate
- Mathieu Ropert