Fault-tolerant Asynchronous Distributed Parallel SGD

CS262a Fall’19

Vinamra Benara (First Year PhD Student, Advisor: Ion Stoica)

Objective

- Robust and **fault tolerant** data parallel distributed training framework for the **cloud**
- Can run seamlessly on preemptible spot instances which are ~90% cheaper than on-demand instances
- No additional overhead upon node failure or node addition (effortless autoscaling)
- Fewer (ideally none) synchronization steps
- Reduced communication
- Immune to **communication** bandwidth and **stragglers**

Problems with Prior Work

- Prior work in distributed training includes Parameter Server and All-Reduce like approaches, ex Push Pull DP-SGD and Horovod.
- These approaches are not amenable for the cloud.
- Work on local setups with high speed comm links.
- No fault tolerance. Halt and restart from scratch upon node failure.
- Costly infrastructure setup
- Extremely sensitive to stragglers
- Frequent costly sync steps
- No peer to peer tensor communication library
- NCCL only supports operations over collectives, ex reduce, gather, broadcast.

Prior Fault Tolerance Methods

- Checkpointing the model params periodically is widely employed to mitigate node failure
- As model sizes are increasing, checkpointing becomes prohibitively expensive.
- Ex. Resnet18 is 11.7M params
- Bert is 340M params
- Checkpointing overhead in results section
- Redundant computation to resume from old checkpoint

Approach

- To achieve the aforementioned objectives, we need both algorithmic and systems support.
- The ideal distributed gradient aggregation would incur
  - Few peer communications
  - Few sync steps
  - Convergence at par with standard frameworks
  - Inherent scalability
  - Resilience to faults
- Stochastic Gradient Push (SGP) is closest to these requirements, yet still far away
- Employs a gossip based PushSUM aggregation

Design with Ray

- Ray is the go-to framework for deployment of large scale compute across clusters.
- I ported SGP to Ray with a clean actor class abstraction.
- The Ray training infrastructure that I designed for SGP, wraps around any standard model.

Architecture

- Ray driver sets up and orchestrates the cluster resources.
- Instantiates actor class instances (workers) and schedules them
- Actor class is a wrapper around SGP and provides all standard methods

Analysis/Benchmarks

Convergence Results with Ray on SGP and All Reduce:

Checkpointing Cost on NFS:

Resume overhead upon fault:

(Checkpointing after every epoch):

(Checkpointing every alternate epoch):

Current Status:

- Working on the making SGP resume without halts in case of node failure. (part of low overhead fault tolerance)