tensorflow-tracing

Performance Tuning in Production

Sayed Hadi Hashemi
Paul Rausch
Benjamin Rabe
Kuan-Yen Chou
Simeng Liu
Volodymyr Kindratenko
Roy H Campbell

May 2019
Performance Changes
Performance Changes

Iteration Time

Time
Performance Changes

Change of Model
Hyper parameters: e.g. Batch Size
Performance Changes

Change of Model
Hyper parameters: e.g. Batch Size
Storage
Network
Memory
Performance Changes

Iteration Time

Time

Change of Model
Hyper parameters: e.g. Batch Size

Storage
Network
Memory

Driver
Software Stack
Misconfiguration
Performance Tuning
Developer

Code Probe
Python
Performance Tuning

Developer

**Code Probe**
- Python

**DAG Probe**
- TensorBoard

Images Credit: Google Brain
Performance Tuning
Developer

**Code Probe**
- Python

**DAG Probe**
- TensorBoard

**Whole DAG Runtime Execution**
- Chrome Tracing

Images Credit: Google Brain
Performance Tuning
Admin

Application-Level
Performance Tuning
Admin

Application-Level

Pros
Effective

Cons
Code Modification
Advance Planning
Could be complicated (e.g. T2T)
Performance Tuning
Admin

**Application-Level**

**Pros**
- Effective

**Cons**
- Code Modification
- Advance Planning
- Could be complicated (e.g. T2T)

**Resource-Level**

**Pros**
- netstat
- nvidia-smi
- NSight
- dstat
- ...

**Cons**
- Too Coarse
- Don’t distinguish different tasks
- The report time is too small
- Data is hard to interpret without context
**Challenges**

Admin

**Detect Problems**
Find the Baseline
Detect Anomaly

**Root Cause Analysis**
Runtime Profiling/Tracing without modification/planning
Data Exchange
**MonkeyPatching**
Intercepts Framework Calls
No need for code modification

**Admin Portal**
Runs at the start of a job
Collects Task-Base Profiling to Establish Baseline
On Demand Tracing/No need for advanced planning

**Tracing File Format**
Portable format
CLI to explore traces
Tensorflow Application

session.run

TensorFlow
**Disabled**
No interception
Only Manage Selected Sessions

**Per Application**
Intercept an application
Manage all the sessions

**System-wide**
Intercept the global library
Manage all applications
Separate Different Tasks
Profile
Collect Automatically
Low Overhead (≈0%)
Establish the Baseline

Trace
Collect On Demand
High Overhead (≈3%)
Root Cause Analysis
Deploy
Campus-wide Deep Learning Cluster
In use at NCSA since Fall 2018

Apache-2
Downloaded +4k times from Pip

Quick Start
pip install tensorflow-tracer

Source Code
https://github.com/xldrx/tensorflow-tracer
Demo
Experiences
Common Causes

**Network**
- Transfer Timing [Hashemi et al, SysML19]
- Congestion
- Wrong Network Interface

**Platform**
- Software Stack
- Drivers
- Containers

**Storage**
- NFS Exhaustion - Rogue Application
- Small Reads vs TFRecords

**Device Placement**
- CPU/GPU
- Locality
Questions
hashemi3@illinois.edu

Quick Start
pip install tensorflow-tracer

Source Code
https://github.com/xldrx/tensorflow-tracer

This work is supported by:
National Science Foundation under Grant No. 1725729