Experienced

Device driver(windows, linux)
Media streaming
CDN
Docker

PyCon Korea 2015 Speaker
PyCon Hongkong 2015 Speaker


python


IP[y]:
IPython


GRPC


NVIDIA
CUDA

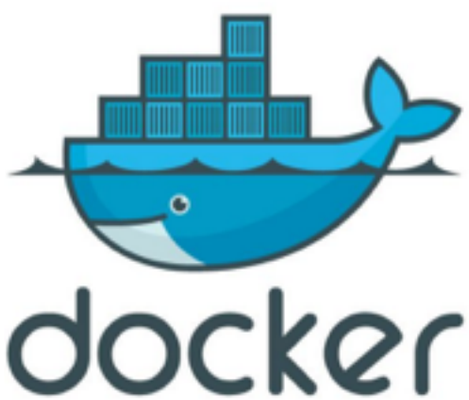

weavescope


YAML


TensorFlow


TensorBoard
EVENTS
IMAGES
GRAPH
HISTOGRAMS
input new regex
Split On Underscores
X Type
STEP
RELATIVE
WALL
Selected Runs:
data
xentropy
xentropy_mean


Google Cloud Platform

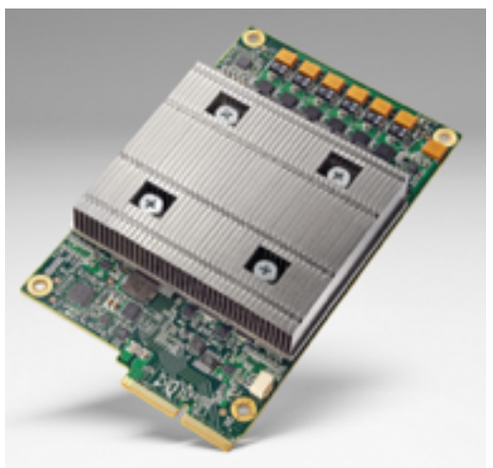

docker


docker-compose

Agenda

- TensorFlow + GPU
- TensorFlow Distributed Environment
- Docker, Google Cloud Platform, AWS..
- Docker, Docker-compose…
- Demo

TensorFlow + GPU



TPU

Different way to install TensorFlow

Pip install

Virtualenv install

Anaconda install

Docker install

Installing from sources

```
// Cool~
$ docker run -it -p 8888:8888 gcr.io/tensorflow/tensorflow

// GPU?
$ path/to/repo/tensorflow/tools/docker/docker_run_gpu.sh
gcr.io/tensorflow/tensorflow:gpu
```

https://github.com/tensorflow/tensorflow,

Branch:master,0.80,0.90rc (Binary download)

- Linux CPU only: Python 2 (build history) / Python 3.4 (build history) / Python 3.5 (build history)

- **Linux GPU**: Python 2 (build history) / Python 3.4 (build history) / Python 3.5 (build history)

- **Mac CPU only**: Python 2 (build history) / Python 3 (build history)

- Android (build history)

// **2)** install & setup for gcc and clang package
$ xcode-select --install

$ vi .zshrc
##########
CUDA
export PATH=/Developer/NVIDIA/CUDA-7.5/bin:$PATH
export DYLD_LIBRARY_PATH=/Developer/NVIDIA/CUDA-7.5/lib:$DYLD_LIBRARY_PATH

$ kextstat | grep -i cuda
  196   0 0xffffff7f82df8000 0x2000   0x2000   com.nvidia.CUDA (1.1.0) 5AFE550D-6361-3897-912D-897C13FF6983 <4 1>

// nvcc compiler
$ nvcc -V
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2015 NVIDIA Corporation
Built on Mon_Apr_11_13:23:40_CDT_2016
Cuda compilation tools, release 7.5, V7.5.26

$ cd /Developer/NVIDIA/CUDA-7.5/samples/

$ sudo make -C 0_Simple/vectorAdd
$ sudo make -C 0_Simple/vectorAddDrv
$ sudo make -C 1_Utilities/**deviceQuery**
$ sudo make -C 1_Utilities/**bandwidthTest**

$ cd bin/x86_64/darwin/release
$ ./deviceQuery
./deviceQuery
./deviceQuery Starting...

CUDA Device Query (Runtime API) version (CUDART static linking)

cudaGetDeviceCount returned 35
-> CUDA driver version is insufficient for CUDA runtime version
Result = FAIL

$ ./bandwidthTest
CUDA Bandwidth Test] - Starting...
Running on...

cudaGetDeviceProperties returned 35
-> CUDA driver version is insufficient for CUDA runtime version
CUDA error at bandwidthTest.cu:255 code=35(cudaErrorInsufficientDriver) "cudaSetDevice(currentDevice)

1) download & install
GPU support : CUDA

https://developer.nvidia.com/cuda-downloads

This Mac
OS X El Capitan
Version 10.11.4
MacBook Pro (Retina, 15-inch, Mid 2015)
Processor 2.2 GHz Intel Core i7
Memory 16GB 1600 MHz DDR3
Graphics Intel Iris Pro 1536 MB

tensorflow / tensorflow

Fork 10,237

Code | Issues 363 | Pull requests 28

Filters ▾ | GPU is:open

New issue

Clear current search query, filters, and sorts

124 Open ✓ 496 Closed

ssignee ▾ Sort ▾

GPU Host To Device copies don't appear to
#2848 opened an hour ago by sjperkins — 0

segfault in perftools::gputools::StreamExecutor::DeviceMemoryUsage - on a busy gpu
#2840 opened 15 hours ago by MInner — 5

Can not completely disable GPU & CUDA support.
#2829 opened 21 hours ago by MrSlayer02 — 2

ERROR when building pip package: undeclared function related to swig
#2811 opened 2 days ago by peterzcc — 1

CUDA_ERROR_MISALIGNED_ADDRESS on MNIST example
#2810 opened 3 days ago by johnfrombluff — 5

Slow quantized graph
#2807 opened 3 days ago by changyun79 — 4

Building TF with custom GCC requires hardcoded ld,nm and as
#2806 opened 3 days ago by akors — 0

error when using batch_normalize in tensorflow.contrib
#2797 opened 3 days ago by guohengkai — 0

Tensorflow hangs without explanation, some threads stay busy. awaiting response
#2788 opened 4 days ago by alexatknit — 10

Potential Contribution: Ansible roles & playbooks
#2780 opened 4 days ago by kuza55 — 0

TensorFlow : Machine Translation example Segmentation fault (Core dumped) awaiting response
#2776 opened 4 days ago by suraj1990 — 2

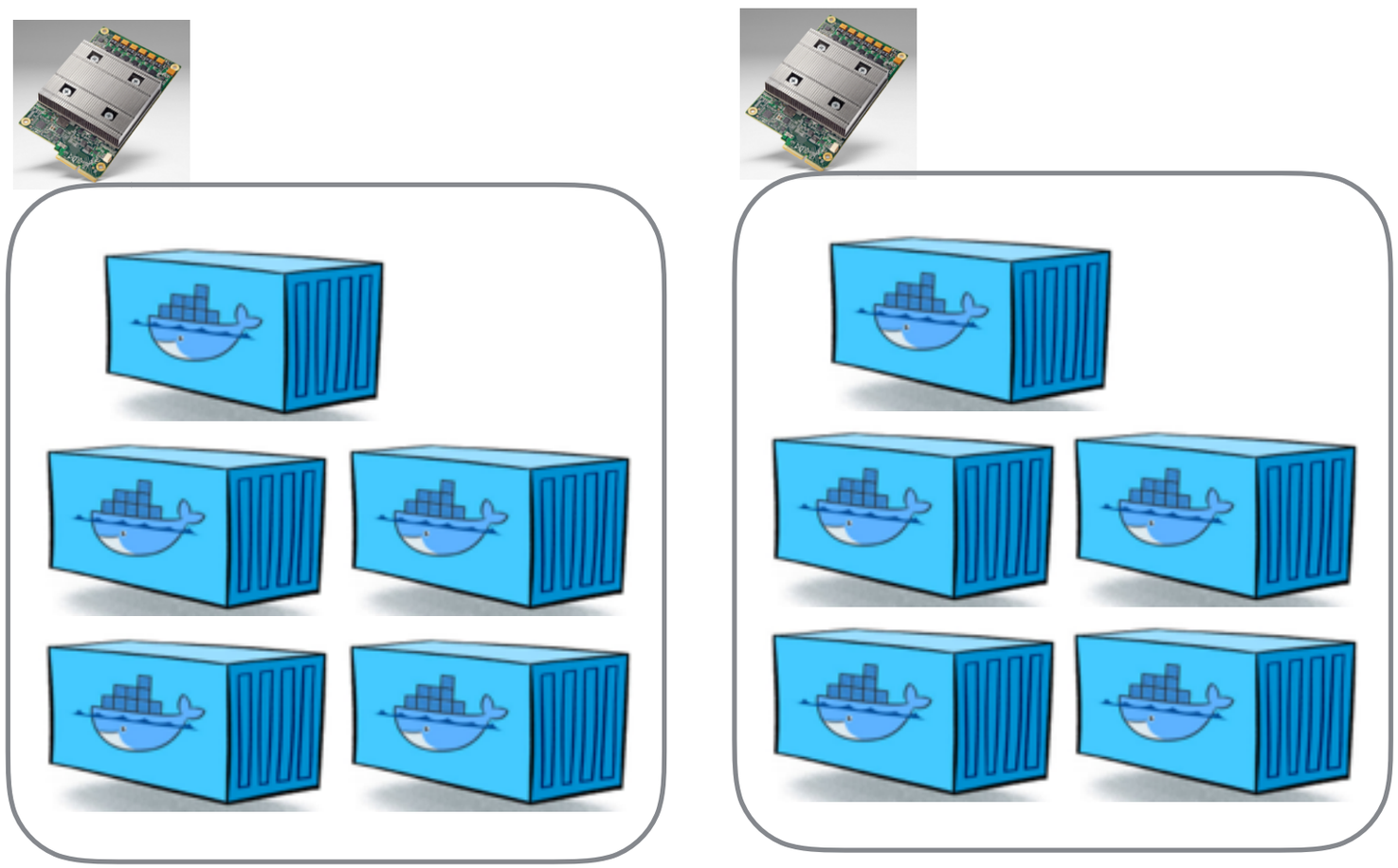
Segmentation fault on tensorflow 0.9.0 awaiting response
#2773 opened 4 days ago by amineHorseman — 1

output ImportError libcudart.so.7.5. in window 7 pycharm awaiting response
#2771 opened 4 days ago by ismymajia — 1

GPU : “Open” issues

blog article :
CUDA+Tensorflow+Docker : https://goo.gl/dIXK2n

Demo for today…

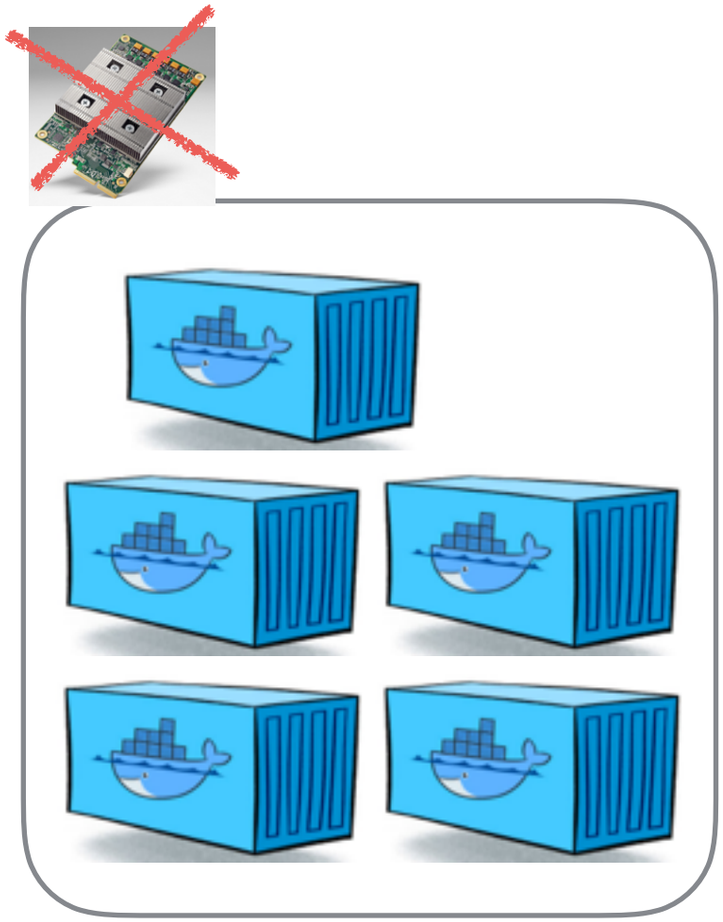


Host MacBook

Host PM2

My Experienced :
FullSeg Player(MPEG-2) + GPU
Media Player + H.264 + GPU
=> Docker+GPU+Tensorflow ???

Demo for today…

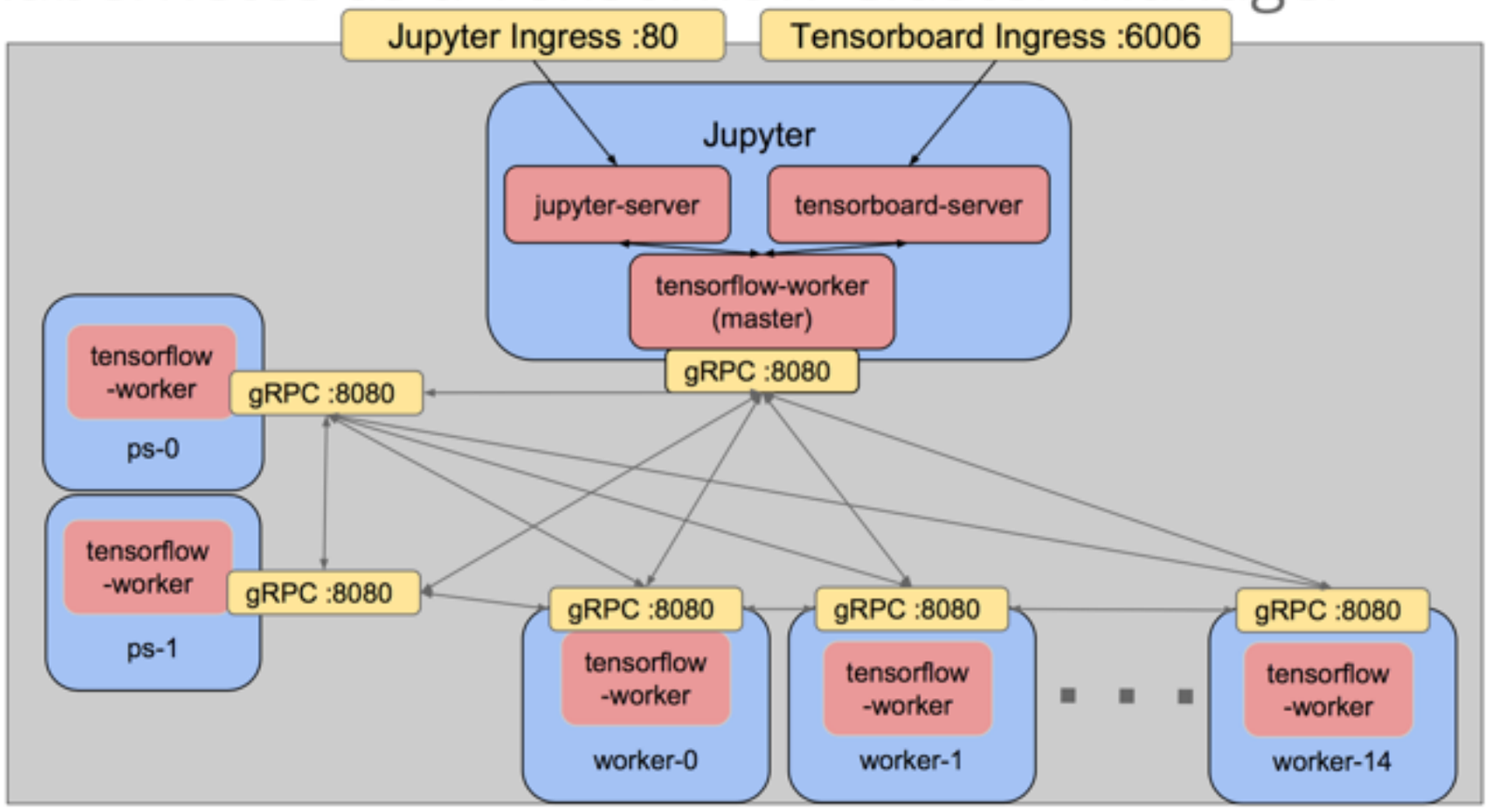


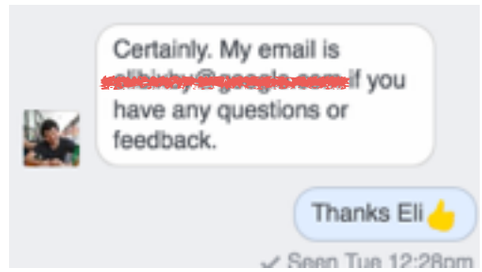
Host MacBook

TensorFlow Distributed Environment

: Basic

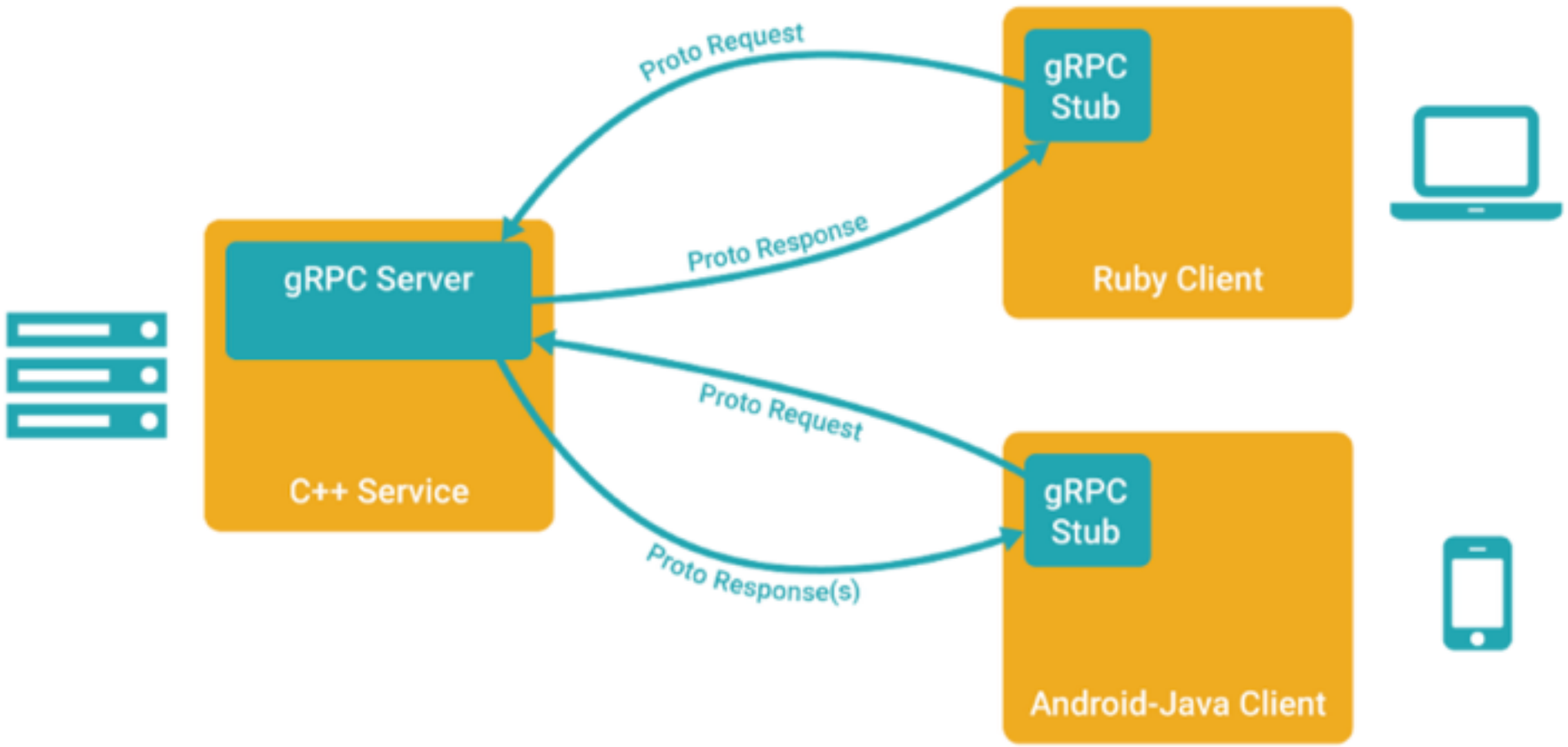
Kubernetes as a Tensorflow Cluster Manager





bit.ly/tensorflow-workshop

gRPC


Proto Request
gRPC Stub
Proto Response
gRPC Server
Ruby Client
Proto Request
gRPC Stub
Proto Response(s)
C++ Service
Android-Java Client

Hello distributed TensorFlow

```
# Start a TensorFlow server
# as a single-process "cluster".
$ python
>>> import tensorflow as tf
>>> c = tf.constant("Hello, distributed TensorFlow!")
>>> server = tf.train.Server.create_local_server()
Initialize HostPortsGrpcChannelCache for job local -> {localhost:55642}
Started server with target: grpc://localhost:55642
>>> sess = tf.Session(server.target)  # Create a session on the server.
>>> sess.run(c)
'Hello, distributed TensorFlow!'
```

class **tf.train.Server**
“ for use in distributed training.”

creates a single-process cluster, with an in-process server.

Create a cluster

cluster
a set of "tasks" that participate in the distributed execution

tf.train.ClusterSpec construction	Available tasks
`tf.train.ClusterSpec({"local": ["localhost:2222", "localhost:2223"]})`	`/job:local/task:0` `/job:local/task:1`

`tf.train.ClusterSpec({`
`  "worker": [`
`  "worker0.example.com:2222",`
`  "worker1.example.com:2222",`
`  "worker2.example.com:2222"`
`  ],`
`  "ps": [`
`  "ps0.example.com:2222",`
`  "ps1.example.com:2222"`
`  ]})`
	`/job:worker/task:0`
	`/job:worker/task:1`
	`/job:worker/task:2`
	`/job:ps/task:0`
	`/job:ps/task:1`

task : worker0 TensorFlow server
…

Create a tf.train.Server instance in each task

Server(cluster, job_name, task_index)
cluster:a set of "tasks" that participate in the distributed execution

```
# In task 0:
cluster = tf.train.ClusterSpec({"local": ["localhost:2222", "localhost:2223"]})
server = tf.train.Server(cluster, job_name="local", task_index=0)
```

```
# In task 1:
cluster = tf.train.ClusterSpec({"local": ["localhost:2222", "localhost:2223"]})
server = tf.train.Server(cluster, job_name="local", task_index=1)
```

cluster -> 1,000 server ?

Specifying distributed devices in your model

```
with tf.device("/job:ps/task:0"):
  weights_1 = tf.Variable(...)
  biases_1 = tf.Variable(...)
```

tf.device() function that is used to specify whether ops run on the CPU or GPU.

```
with tf.device("/job:worker/task:7"):
  input, labels = ...
  layer_1 = tf.nn.relu(tf.matmul(input, weights_1) + biases_1)
  logits = tf.nn.relu(tf.matmul(layer_1, weights_2) + biases_2)
  # ...
  train_op = ...
```

TensorFlow will insert the appropriate data transfers between the jobs(ps->worker, worker->ps)

```
with tf.Session("grpc://worker7.example.com:2222") as sess:
  for _ in range(10000):
    sess.run(train_op)
```

Docker,
Google Cloud Platform,
AWS ..

TensorFlow

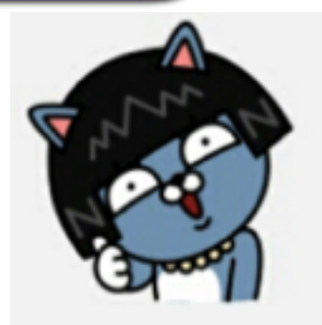
GPU
Distributed Env.
Docker
Google Cloud
AWS
Azure
Native PM

$$$
GPU

Distributed Env.
Docker
Google Cloud
AWS
Azure
Native PM

$$$
AWS
Google Cloud
Azure
Native PM(+TPU)

Distributed Env.
Docker

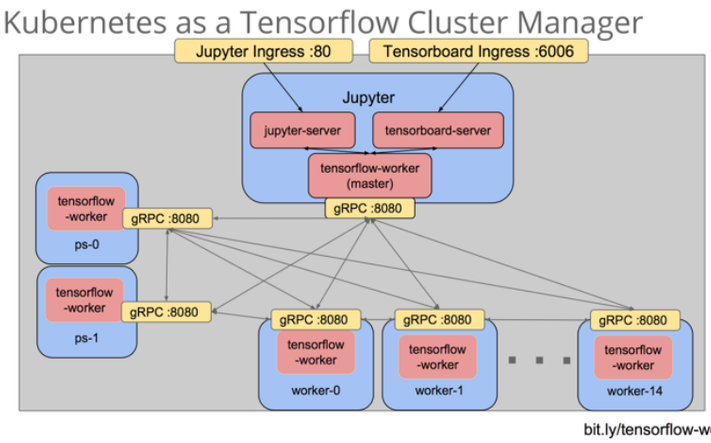


http://104.155.200.100/text1.zip

Compute Engine

Jupyter-server
(192.168.99.101:8888)

Tensorboard-server
(192.168.99.101:6006)



tensorflow-worker :
master-0

gRPC:8080

tensorflow-worker :
ps-0

gRPC:8080

gRPC:8080

tensorflow-worker :
ps-1

gRPC:8080

gRPC:8080

gRPC:8080

weavescope
192.168.99.101:404

tensorflow-worker :
work-0

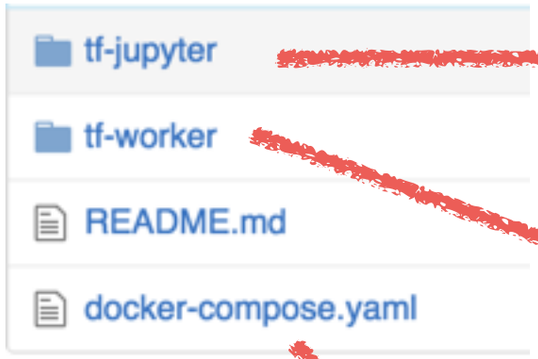
tensorflow-worker :
work-1

tensorflow-worker :
work-14

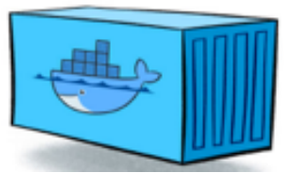
Docker, Docker-compose…

Demo code :
https://github.com/bwahn/tensorflow-kr-docker

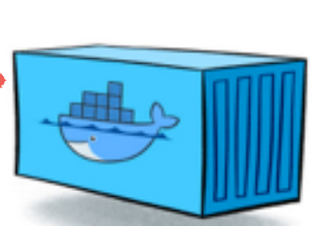
github:



Jupyter-server



tensorflow-worker : (ps-0/1/2, worker-0/1/.., master-0)



compose file for Demo

tf-worker

..

Dockerfile

worker.py

```
FROM gcr.io/tensorflow/tensorflow:latest-devel
ADD worker.py worker.py
CMD ["python", "worker.py"]
```

```
import tensorflow as tf
import os
import logging
import sys
import ast

root = logging.getLogger()
root.setLevel(logging.INFO)
ch = logging.StreamHandler(sys.stdout)
root.addHandler(ch)

POD_NAME = os.environ.get('POD_NAME')
logging.info(POD_NAME)
```

role defined

```
def main(job_name, task_id):
    server = tf.train.Server(
        {
            "ps": [
                "ps-0:8080",
                "ps-1:8080",
                "ps-2:8080",
                "ps-3:8080",
            ],
            "worker": [
                "worker-0:8080",
                "worker-1:8080",
                "worker-2:8080",
                "worker-3:8080",
                "worker-4:8080",
                "worker-5:8080",
                "worker-6:8080",
                "worker-7:8080",
            ],
            "master": [
                "master-0:8080",
            ],
        },
        job_name=job_name,
        task_index=task_id
    )
    server.join()


if __name__ == '__main__':
    this_job_name, this_task_id = POD_NAME.split('-', 2)
    main(this_job_name, int(this_task_id))
```

ps-0 (docker-compose.yaml)

```
version: '2'
services:
####
  ps-0:
    build: ./tf-worker/
    container_name: ps-0
    image: ps-0:0.1
    ports:
      - "8080"
    environment:
      POD_NAME: ps-0
....
```

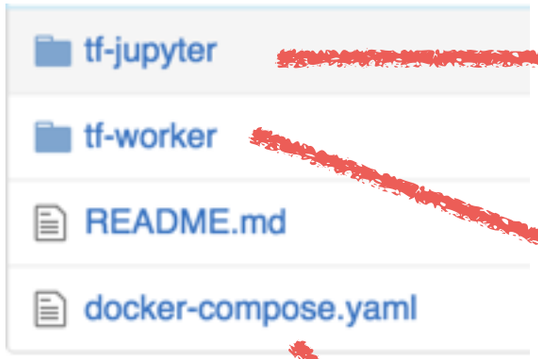
gRPC port

POD_NAME env in worker.py

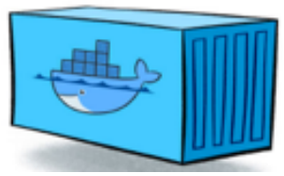
```
$ docker-compose -f docker-compose.yaml build ps-0
Building ps-0
Step 1 : FROM gcr.io/tensorflow/tensorflow:latest-devel
 ---> c3efccc5f94f
Step 2 : ADD worker.py worker.py
 ---> Using cache
 ---> f9c2d840b051
Step 3 : CMD python worker.py
 ---> Using cache
 ---> 57a6c024580c
Successfully built 57a6c024580c
$ docker images | grep ps -0
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
ps-0                0.1                 57a6c024580c        2 days ago          2.38 GB
```

```
//docker-compose.yaml

version: '2'
services:
####
  ps-0:
    build: ./tf-worker/
    container_name: ps-0
    image: ps-0:0.1
    ports:
      - "8080"
    environment:
      POD_NAME: ps-0

  ps-1:
    build: ./tf-worker/
    container_name: ps-1
    image: ps-1:0.1
    ports:
      - "8080"
    environment:
      POD_NAME: ps-1

  ps-2:
    build: ./tf-worker/
    container_name: ps-2
    image: ps-2:0.1
    ports:
      - "8080"
    environment:
      POD_NAME: ps-2

  ps-3:
    build: ./tf-worker/
    container_name: ps-3
    image: ps-3:0.1
    ports:
      - "8080"
    environment:
      POD_NAME: ps-3
####
  worker-0:
    build: ./tf-worker/
    container_name: worker-0
    image: worker-0:0.1
    ports:
      - "8080"
    environment:
      POD_NAME: worker-0

  worker-1:
    build: ./tf-worker/
    container_name: worker-1
    image: worker-1:0.1
    ports:
      - "8080"
    environment:
      POD_NAME: worker-1

  worker-2:
    build: ./tf-worker/
    container_name: worker-2
    image: worker-2:0.1
    ports:
      - "8080"
    environment:
      POD_NAME: worker-2

  worker-3:
    build: ./tf-worker/
    container_name: worker-3
    image: worker-3:0.1
    ports:
      - "8080"
    environment:
      POD_NAME: worker-3

  worker-4:
    build: ./tf-worker/
    container_name: worker-4
    image: worker-4:0.1
    ports:
      - "8080"
    environment:
      POD_NAME: worker-4

  worker-5:
    build: ./tf-worker/
    container_name: worker-5
    image: worker-5:0.1
    ports:
      - "8080"
    environment:
      POD_NAME: worker-5

  worker-6:
    build: ./tf-worker/
    container_name: worker-6
    image: worker-6:0.1
    ports:
      - "8080"
    environment:
      POD_NAME: worker-6

  worker-7:
    build: ./tf-worker/
    container_name: worker-7
    image: worker-7:0.1
    ports:
      - "8080"
    environment:
      POD_NAME: worker-7

####
  master-0:
    build: ./tf-worker/
    container_name: master-0
    image: master-0:0.1
    ports:
      - "8080"
    environment:
      POD_NAME: master-0
```

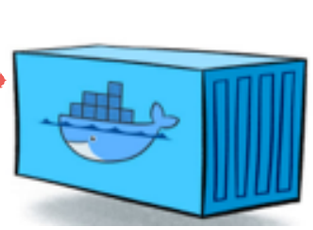
github:



Jupyter-server



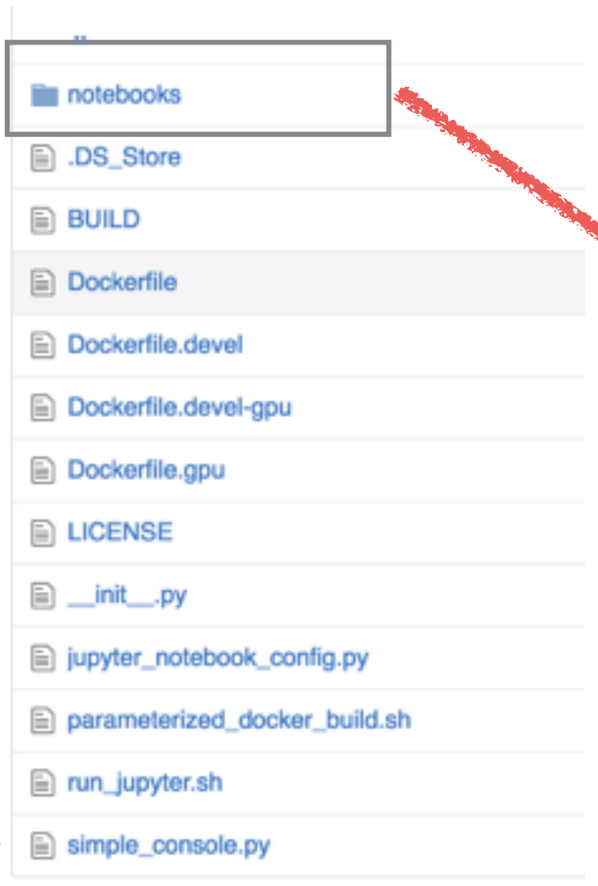
tensorflow-worker : (ps-0/1/2, worker-0/1/.., master-0)

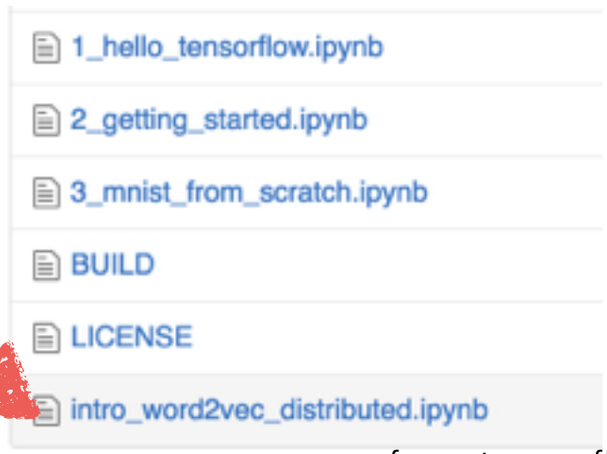


compose file for Demo

tf-jupyter

```
// TensorFlow github
$ git clone https://github.com/tensorflow/tensorflow.git
$ cd /tensorflow-git/tensorflow/tools/docker
```





from tensorflow-workshop

```
$ docker-compose -f docker-compose.yaml build jupyter-server
Building jupyter-server
Step 1 : FROM ubuntu:14.04
 ---> 8f1bd21bd25c
Step 2 : MAINTAINER Craig Citro <craigcitro@google.com>
 ---> Using cache
 ---> c87f53e1cfc9
Step 3 : RUN apt-get update && apt-get install -y curl libfreetype6-dev libpng12-dev
python-scipy && apt-get clean && rm -rf /var/lib/apt/lists/*
 ---> Using cache
 ---> a5abc6a5ed5a
Step 4 : RUN curl -O https://bootstrap.pypa.io/get-pip.py && python get-pip.py && rm get-pip.
 ---> Using cache
 ---> 39df3de28486
Step 5 : RUN pip --no-cache-dir install ipykernel jupyter matplotlib && python
 ---> Using cache
 ---> ce0e0004e0a6
Step 6 : ENV TENSORFLOW_VERSION 0.8.0
 ---> Using cache
 ---> 80791608c082
Step 7 : RUN pip --no-cache-dir install http://storage.googleapis.com/tensorflow/linux/cpu/tensorf
 ---> Using cache
 ---> 980ebdfed88c
Step 8 : COPY jupyter_notebook_config.py /root/.jupyter/
 ---> Using cache
 ---> 2e2aa264d165
Step 9 : COPY notebooks /notebooks
 ---> Using cache
 ---> 3b6409cc98a9
Step 10 : COPY run_jupyter.sh /
 ---> Using cache
 ---> b6cfb46577ed
Step 11 : EXPOSE 6006
 ---> Using cache
 ---> e4010cf837c6
Step 12 : EXPOSE 8888
 ---> Using cache
 ---> 0266ba056034
Step 13 : WORKDIR "/notebooks"
 ---> Using cache
 ---> 48983eb7b7af
Step 14 : CMD /run_jupyter.sh
 ---> Using cache
 ---> 364efc0d61ce
Successfully built 364efc0d61ce

$ docker images | grep tf-jupyter-server
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
tf-jupyter-server   0.1                 364efc0d61ce        4 days ago          722 MB
```

```
//docker-compose.yaml
…
  jupyter-server:
    build:
      context: ./tf-jupyter/
    container_name: tf-jupyter-server
    image: tf-jupyter-server:0.1
    ports:
      - "8888:8888"
      - "80:80"
    volumes:
      - /tmp/tf/tensorflow-logs:/var/log/tensorflow
```

http://104.155.200.100/text1.zip

Compute Engine

Jupyter-server
(192.168.99.101:8888)

Tensorboard-server
(192.168.99.101:6006)

tensorflow-worker :
master-0

gRPC:8080

tensorflow-worker :
ps-0

gRPC:8080

tensorflow-worker :
ps-1

gRPC:8080

gRPC:8080

gRPC:8080

tensorflow-worker :
work-0

tensorflow-worker :
work-1

tensorflow-worker :
work-14

Tensorboard-server

```
//docker-compose.yaml
…
  tf-tensorboard-server:
    image: gcr.io/tensorflow/tensorflow:latest-devel
    container_name: tf-tensorboard-server
    ports:
      - "6006:6006"
    command: /tensorflow/bazel-bin/tensorflow/tensorboard/tensorboard --logdir=/var/log/tensorflow
    volumes:
      - /tmp/tf/tensorflow-logs:/var/log/tensorflow
```

Summary
- Tensorflow Distributed Environment (with Docker)

Future
- Native Clustering (several hosts): Docker Swarm, RancherOS
- Docker + GPU Architecture & Test Environment

Q&A