TensorFlowOnSpark

Scaled ML@Stanford

March 25, 2017

Machine Learning, AI & No Free Lunch

- Five key ingredients for ML towards AI
  1. Lots & lots of data
  2. Very flexible models
  3. Enough computing power
  4. Computationally efficient inference
  5. **Powerful priors that can defeat the curse of dimensionality**

* Yoshua Bengio @ ICDM 2016

What is TensorFlowOnSpark?

YAHOO!

What's TensorFlowOnSpark?

- Scale up TensorFlow apps with minimal changes
- Support all TensorFlow functionalities
  - Model/data parallelism, Synch/Asynch, TensorBoard
- Integrate with existing data & pipeline
  - ex. HDFS, SQL, MLlib
- Deployable on cloud or on-premise

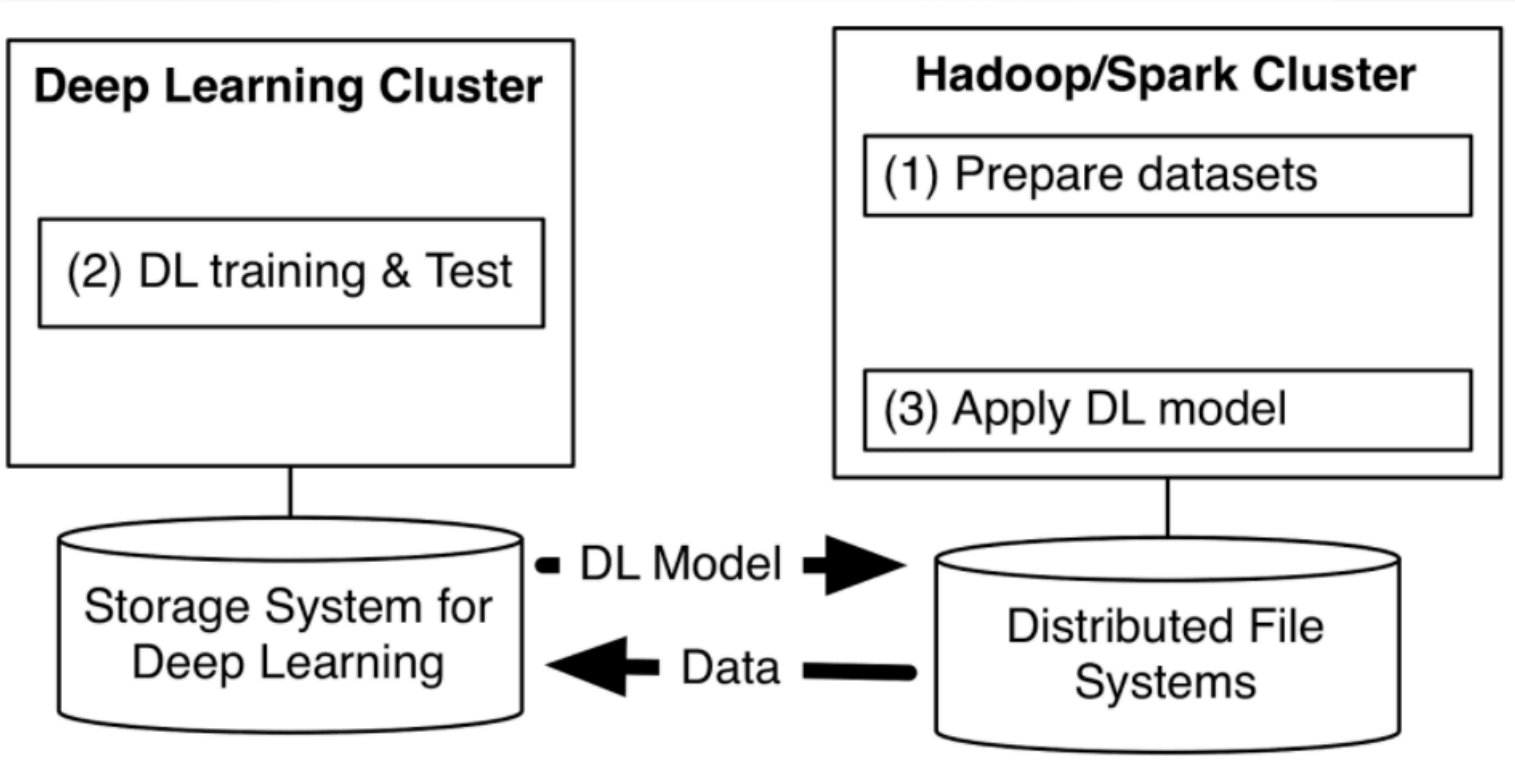
YAHOO!

Why TensorFlowOnSpark at Yahoo?

- Major player of open-source ecosystem
  - Birth place of Apache Hadoop
  - Adopter/contributor of Spark since 2013
- Large clusters in house
  - Tens of clusters
  - Thousands of nodes per cluster
- Massive amount of data
  - Petabytes of data

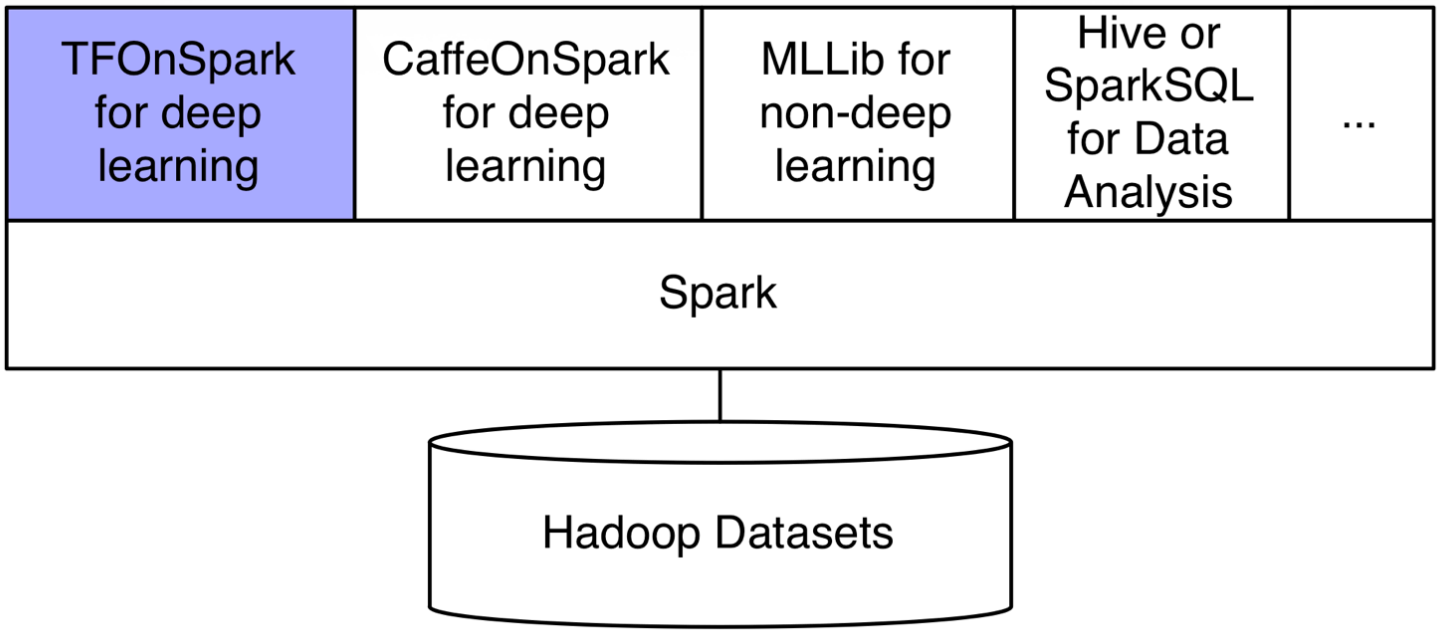
YAHOO!

Why TensorFlowOnSpark?



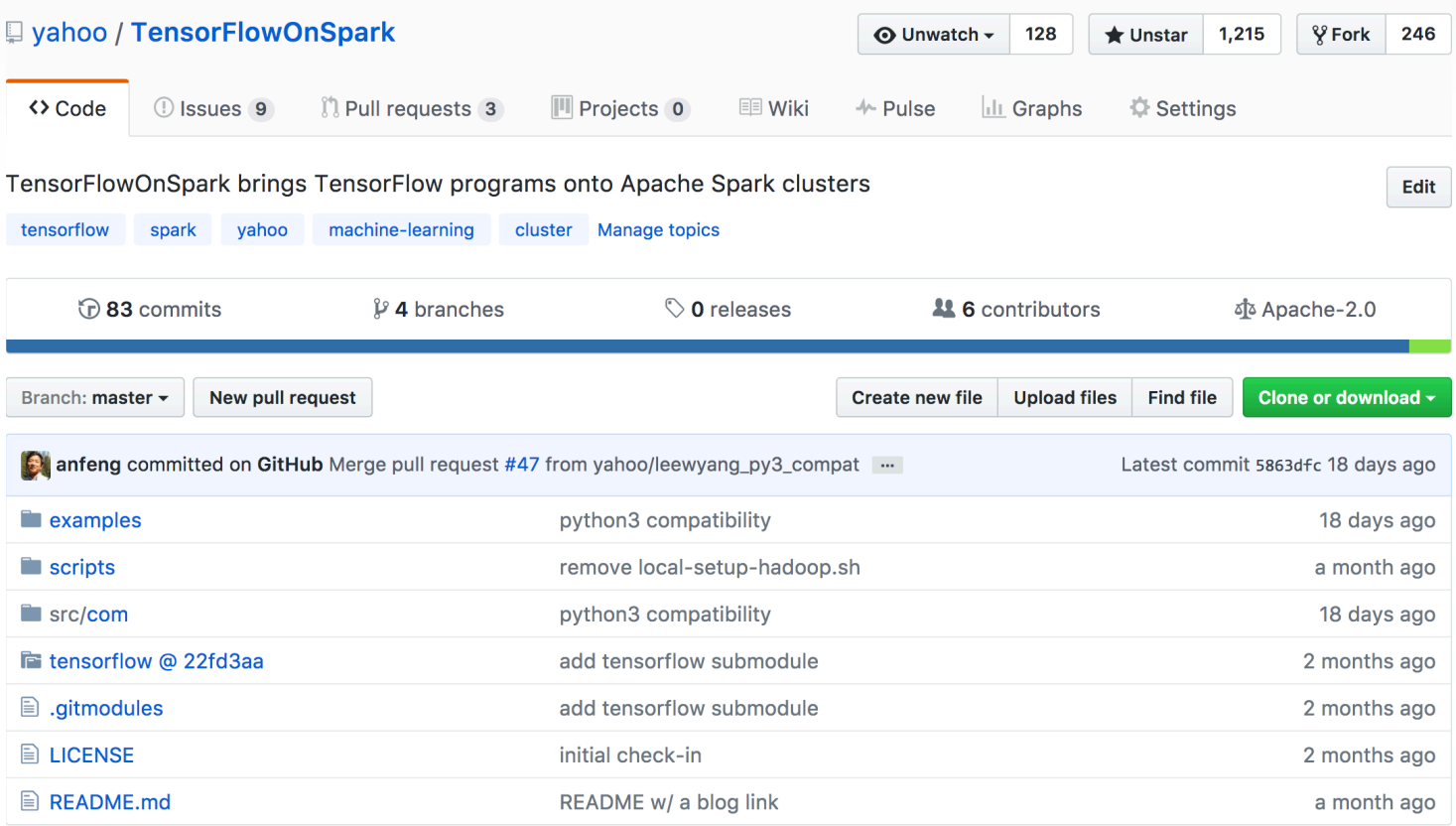
YAHOO!

TensorFlowOnSpark



YAHOO!

Open Source: github.com/yahoo/TensorFlowOnSpark



YAHOO!

TensorFlowOnSpark

- Launches TF clusters using Spark executors
- Supports TF data ingestion modes
  - Spark – RDD.mapPartitions()
  - TenforFlow – directly access HDFS
- Supports TensorBoard during/after training
- Generally agnostic to Spark/TF versions

YAHOO!

TFoS Basics

1. **Launch** TensorFlow cluster
2. **Feed data** to TensorFlow app
3. **Shutdown** TensorFlow cluster

YAHOO!

TFoS Python API

```
cluster = TFCluster.run(sc, map_fn, args, num_executors,
num_ps, tensorboard, input_mode)

cluster.train(dataRDD, num_epochs=0)

cluster.inference(dataRDD)

cluster.shutdown()
```

YAHOO!

TFoS: Minimum Code Changes

```
# diff -w eval_image_classifier.py
20a21,27
> from pyspark.context import SparkContext
> from pyspark.conf import SparkConf
> from com.yahoo.ml.tf import TFCluster, TFNode
> import sys
>
> def main_fun(argv, ctx):
27a35,36
>   sys.argv = argv
84,85d92
< def main(_):
88a96,97
>   cluster_spec, server = TFNode.start_cluster_server(ctx)
191c200,204
<   tf.app.run()
---
>   sc = SparkContext(conf=SparkConf().setAppName("eval_image_classifier"))
>   num_executors = int(sc._conf.get("spark.executor.instances"))
>   cluster = TFCluster.run(sc, main_fun, sys.argv, num_executors, 0, False, TFCluster.InputMode.TENSORFLOW)
>   cluster.shutdown()
```

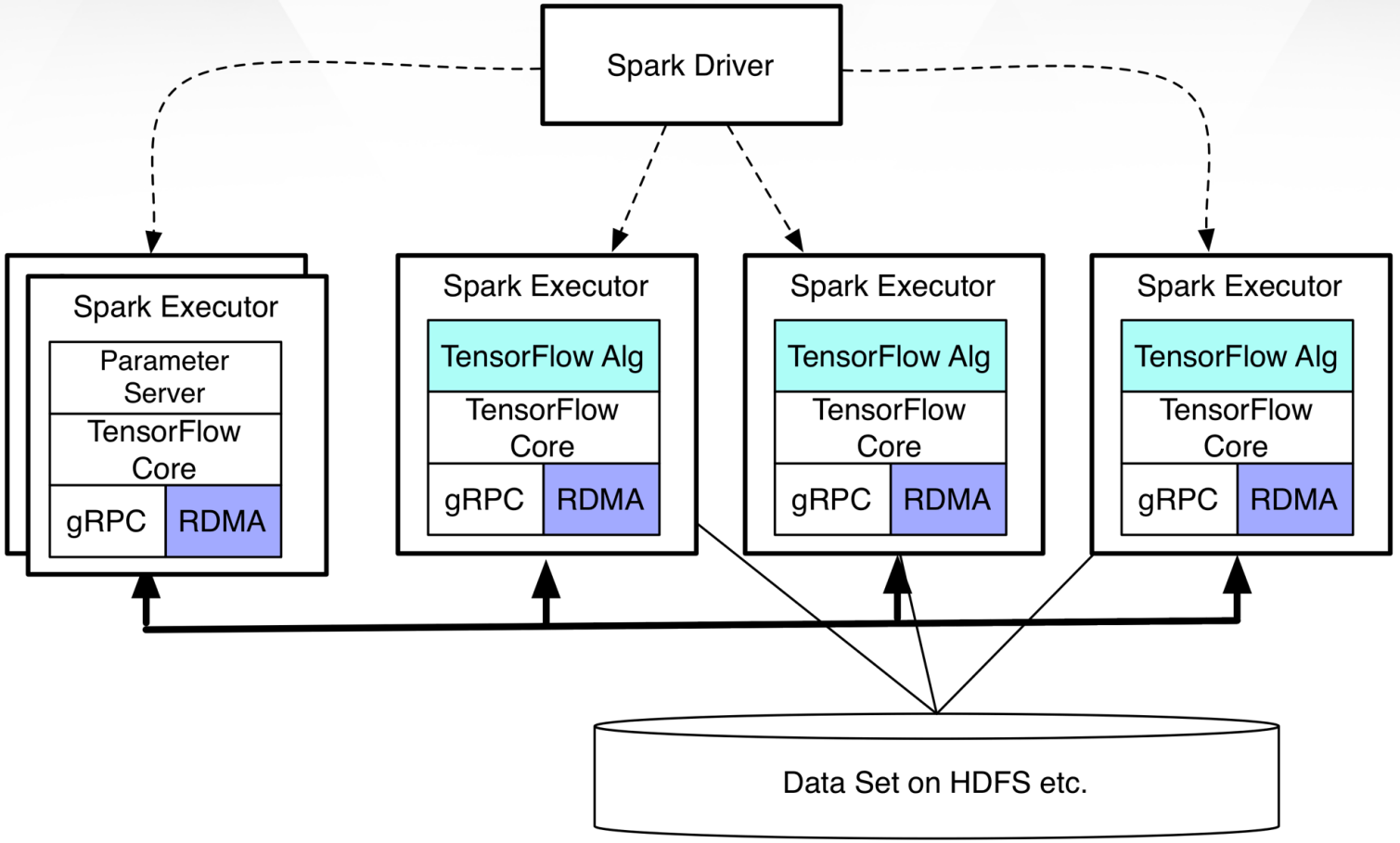
YAHOO!

TFoS Input Modes

- InputMode.SPARK
  - feed_dict
  - Small-medium scale data
  - Fed via RDD.mapPartitions()
- InputMode.TENSORFLOW
  - Reader + QueueRunner
  - Large scale data
  - Reads directly from HDFS

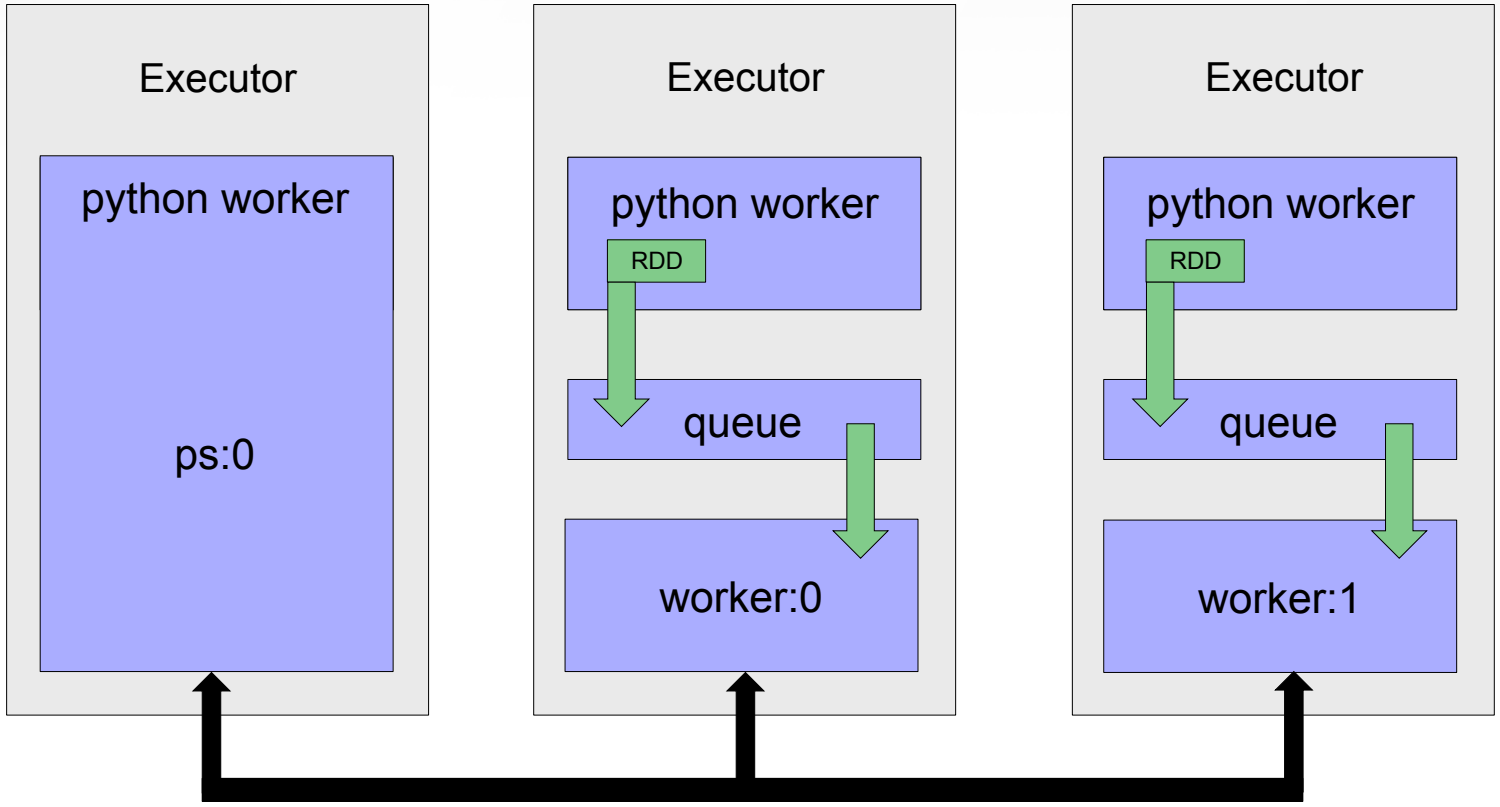
YAHOO!

TFoS Architecture



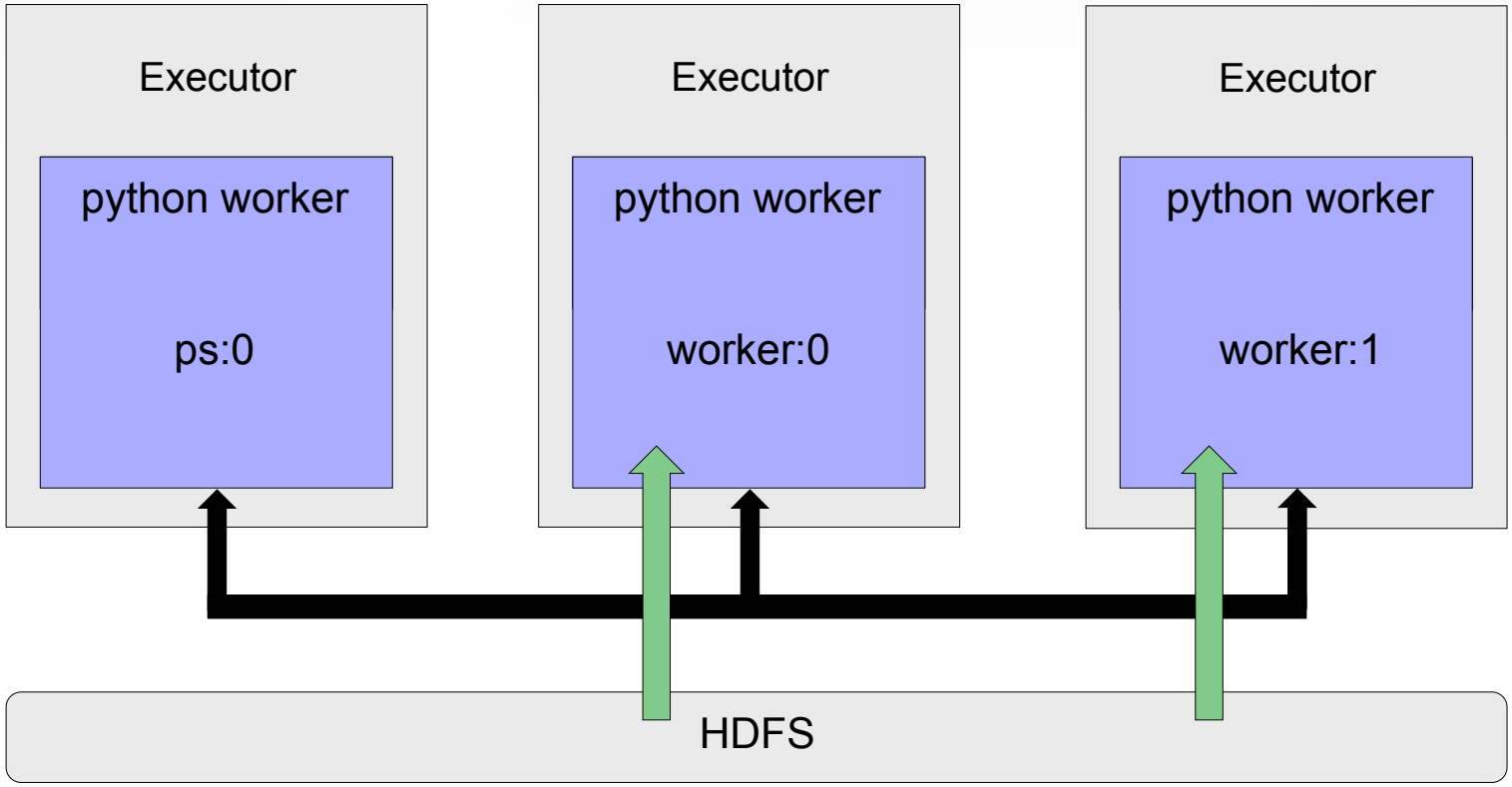
YAHOO!

TFoS: InputMode.SPARK



YAHOO!

TFoS: InputMode.TENSORFLOW



YAHOO!

TFoS: Failure Recovery

- TF Checkpoints written to HDFS
- InputMode.SPARK
  - TF worker runs in background
  - RDD data feeding tasks can be retried
  - However, TF worker failures will be “hidden” from Spark
- InputMode.TENSORFLOW
  - TF worker runs in foreground
  - TF worker failures will be retried as Spark task
  - TF worker restores from checkpoint

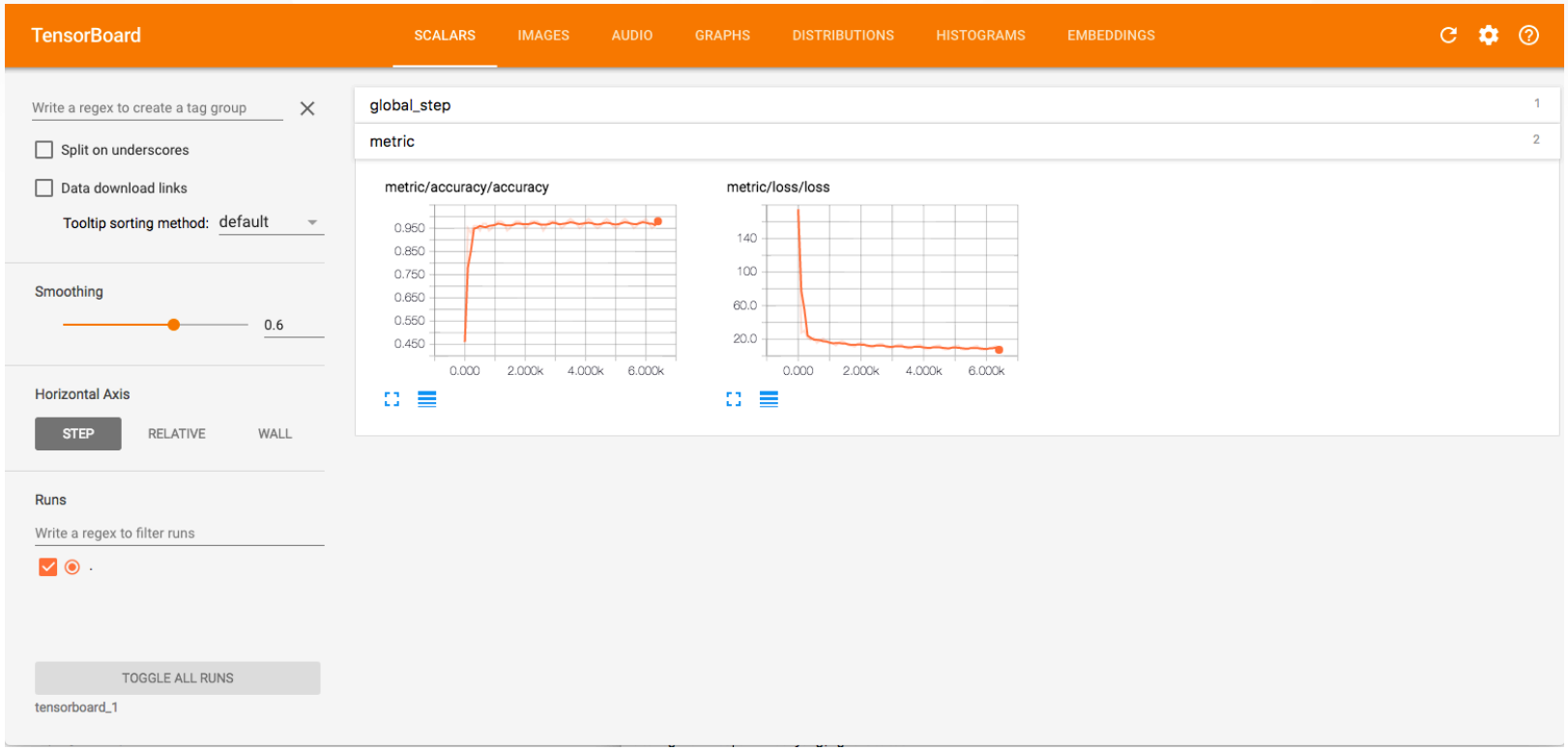
YAHOO!

TFoS: Failure Recovery

- Executor failures are problematic
  - TF cluster_spec is statically-defined
  - YARN doesn't re-allocate on same node
  - Port may no longer be available
- Need dynamic cluster membership
  - Explore options w/ TensorFlow team

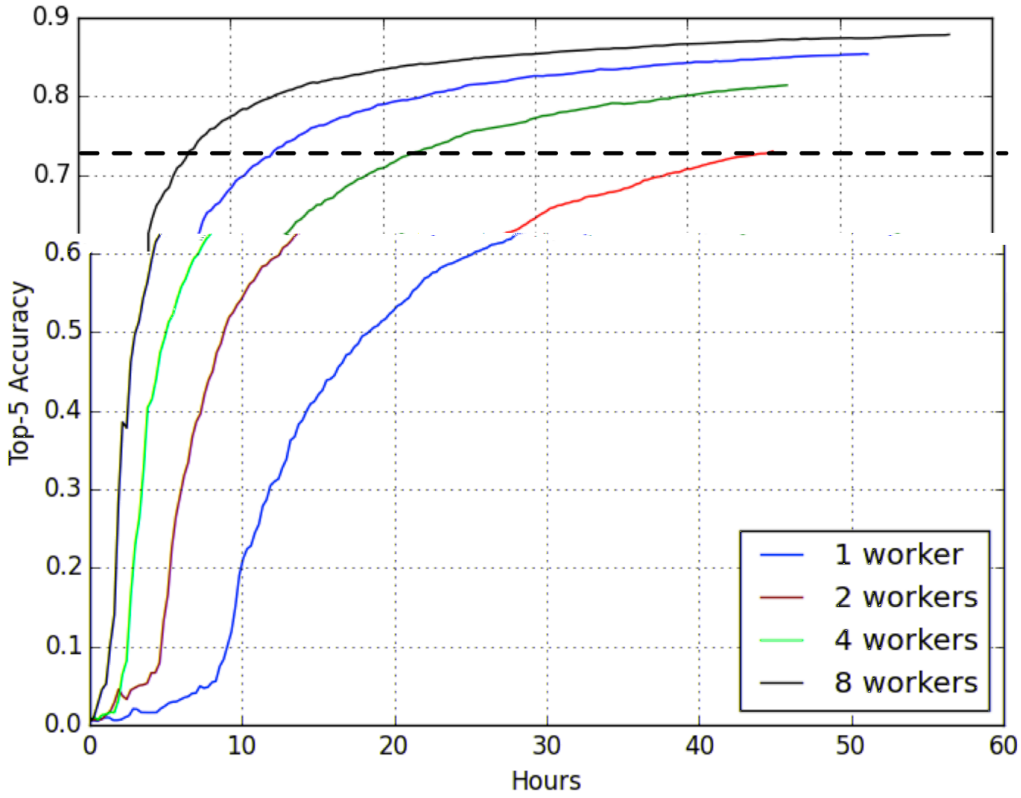
YAHOO!

TensorBoard on TFoS



YAHOO!

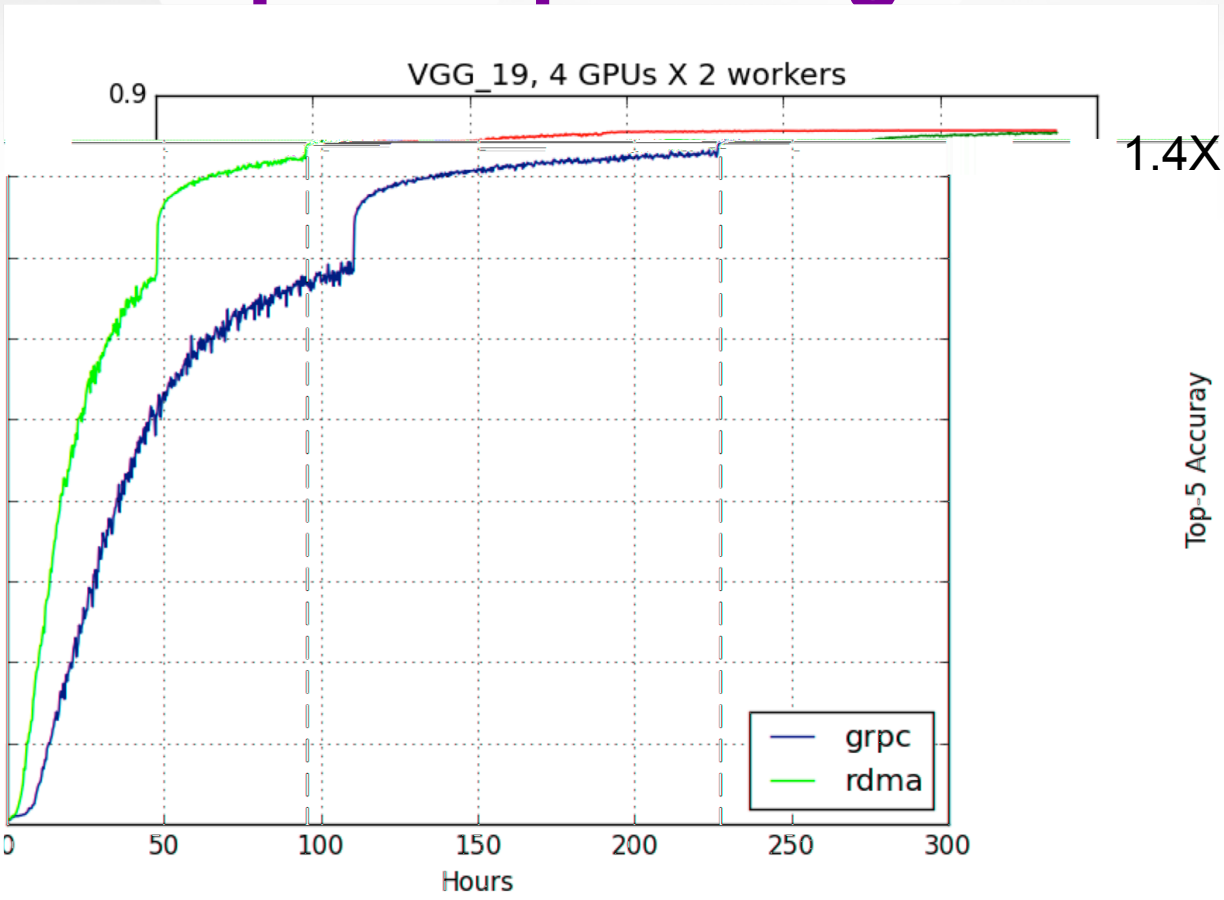
TFoS Scaling



Near linear scalability

YAHOO!

RDMA Speedup over gRPC



1.4X speedup

YAHOO!

Related Work

	SparkNet	TensorFrames	TFonS
Programming Language	Scala	Python, Scala	Python
Migration	Major	Medium	Minor
Parallelism	Data Parallelism	Data Parallelism	Data + Model Parallelism
Distributed Training	Synchronous	Synchronous	Synchronous + Asynchrnous
TensorBoard	✗	✗	✓
Scalability	Driver bottleneck	Driver bottleneck	✓

Summary

- TFoS brings deep learning to big-data clusters
  - TensorFlow: 0.12 -1.0
  - Spark: 1.6-2.x
  - Cluster manager: YARN, Standalone, Mesos
  - EC2 image provided
- RDMA enhancement for faster training
  - PR for github/tensorflow repo

YAHOO!

Questions?

https://github.com/yahoo/TensorFlowOnSpark