Distributed TensorFlow on Hops

Fabio Buso (@siroibaf)
Head of Engineering – Logical Clocks AB

TensorFlow Day – OSCON 2018
[...]

the general consensus seems to be that everyone expects some gain in performance numbers if the dataset size is increased dramatically [...]

Sun et Al. - Revisiting Unreasonable Effectiveness of Data in Deep Learning Era - 2017

Joel et Al. - Deep Learning Scaling is Predictable, Empirically - 2017
Big data needs scalable storage: HopsFs

- Metadata stored in distributed, in-memory db
  - 37x increased capacity
- Multiple active namenodes
  - 16x increased throughput
- Native support for HDFS/HopsFS in TensorFlow
HopsFS support for Small Files

- Open Images Dataset:
  - 9m images
  - ~80% small files (<64 KB)
- Reading dataset takes ~ 2 minutes
  - 5X performance improvement
  - 75K images/seconds
Data security in Hops

- X.509 Certificates for authentication
  - 1 Certificate for each project user
  - New App certificate generated for each job

- Store an audit trail of the operations (read/write/create/etc) users and apps perform on HopsFs
The control plane

Hopsworks
Rest API

LDAP
Active Directory
Spark
kafka
Flink
jupyter
Hive

Hops
elastic
kibana
InfluxDB
Grafana

Logical Clocks
Experiments in Hops
“I have to run a hundred experiments to find the best model,” he complained, as he showed me his Jupyter notebooks. “That takes time. Every experiment takes a lot of programming, because there are so many different parameters.

[https://thomaswdinsmore.com/2018/01/30/predictions-for-2018/ ]
Experiments in TensorFlow/Hops

- Run and evaluate multiple models in parallel on a subset of the dataset
- 3 execution modes:
  - Basic mode
  - Grid search
  - Differential Evolution
Differential evolution search

Steps:
1. Pick a random population
2. Evaluate the population
3. Modify population with mutations and crossovers
4. Repeat steps 2 - 3
Differential Evolution search (TensorFlow/Hops)

def model(lr, dropout):
    ...

boundaries = {
    'learning_rate': [0.001, 0.0005],
    'dropout': [0.45, 0.9]}

experiments.evolutionary_search(spark, model, boundaries, direction='max', generations=10, popsize=10, mutation=0.5, crossover=0.7)
Demo
Reproducible experiments

- Results tracking
- Hyperparameter tracking
- Jupyter notebook versioning
- Conda Env versioning
- WIP: Dataset versioning
Distributed Training
Distributed TensorFlow

- Needs to scale to multiple GPUs and Machines
- TF Estimator API → Start multiple processes and set TF_CONFIG
  - Use a cluster manager
- 2 architectures:
  - Parameter Server
  - Ring All-Reduce.
TensorFlowOnSpark (TFoS) by Yahoo!

- Distributed TensorFlow over Spark
- PS/Workers executed inside Spark executors
- Uses Spark to allocate resources and distribute code
- Manages Tensorboard
Run TFoS (TensorFlow/Hops)

```python
def training_fun(argv, ctx):
    .....  
    TFNode.start_cluster_server()
    .....  
    TFCluster.run(spark, training_fun, num_exec, num_ps...)
```

Full conversion guide:
All Reduce architecture
PS server architecture doesn’t scale

* Ermias Gebremeskel - Analysis and Comparison of Distributed Training for Deep Neural Networks in a Dynamic Environment.
Horovod by Uber

- Built on Nvidia NCCL2
- Gradient updates distributed using Ring All-Reduce Algorithm
- Increase network I/O during back-propagation by sharing updates at higher layers in parallel with computing weight updates at lower layers
GPU utilization – Horovod vs PS

* Ermias Gebremeskel - Analysis and Comparison of Distributed Training for Deep Neural Networks in a Dynamic Environment - 2018
Multi-Machine Horovod on Hops

Spark Driver

Allocate resources

MPI Wrapper

Monitor

Setup monitor

MPI RUN

Start MPI processes

Executor

MPI

Executor

MPI

Executor

MPI

Executor

MPI
import horovod.tensorflow as hvd

def conv_model(feature, target, mode):
    ....

def main(_):
    hvd.init()
    opt = hvd.DistributedOptimizer(opt)
    ....

from hops import allreduce
allreduce.launch(spark, 'hdfs:///Projects/.../note.ipynb')
Demo
Model Serving
Scale model serving with Kubernetes

Considered best practice by the community

Provide tools to easily:
- Fault tolerance
- Rolling release new models
- Autoscaling
Production system

Model monitoring infrastructure

Re-train and deploy new model

HopsFS

Serving infrastructure