Distributed Deep Learning Inference
using Apache MXNet* and Apache Spark

Naveen Swamy
Amazon AI
Outline

• Review of Deep Learning

• Apache MXNet Framework

• Distributed Inference using MXNet and Spark
Deep Learning

- Originally inspired by our biological neural systems.
- A System that learns important features from experience.
- Layers of Neurons learning concepts.
- Deep learning != deep understanding

Credit: Ian Goodfellow et al., Deep Learning Book
Algorithmic Advances (Faster Learning)

Abundance of Data (Deeper Networks)

High Performance Compute
GPUs (Faster Experiments)

Bigger and Better Models = Better AI Products
Why does Deep Learning matter?

- Personal Assistants
- Autonomous Vehicles
- Health care
- Solve Intelligence???
Deep Learning & AI, Limitations

DL Limitations:

• Requires lots of data and compute power.

• Cannot detect Inherent bias in data - Transparency.

• Uninterpretable Results.
Data

- Pass data through the network – forward pass
- Define an objective – Loss function
- Send the error back – backward pass

Model: Output of Training a neural network
Deep Learning Inference

- **Real time Inference**: Tasks that require immediate result.

- **Batch Inference**: Tasks where you need to run on a large data sets.
  - Pre-computations are necessary - Recommender Systems.
  - Backfilling with state-of-the art models.
  - Testing new models on historic data.
Types of Learning

• **Supervised Learning** – Uses labeled training data learning to associate input data to output.
  Example: Image classification, Speech Recognition, Machine translation

• **Unsupervised Learning** - Learns patterns from Unlabeled data.
  Example: Clustering, Association discovery.

• **Active Learning** – Semi-supervised, human in the middle.

• **Reinforcement Learning** – learn from environment, using feedback.
Outline

• Review of Deep Learning

• Apache MXNet Framework

• Distributed Inference using MXNet and Spark
Why MXNet

- Programmable
  - Simple Syntax
  - Imperative/Declarative
  - Multiple languages

- Portable
  - Highly efficient models
  - for Mobile and IOT

- High Performance
  - Near linear scaling across hundreds of GPUs

- Open Source
  - Incubating at Apache

ONNX Support

Gluon: Easily and quickly build high performance models with Imperative APIs
MXNet – NDArray & Symbol

- **NDArray** – Imperative Tensor Operations that work on both CPU and GPUs.
- **Symbol APIs** – similar to NDArray but adopts declarative programming for optimization.

![Symbolic Program](Image1)

![Computation Graph](Image2)
MXNet - Module

High level APIs to work with Symbol

1) Create Graph

```python
>>> data = mx.sym.Variable('data')
>>> fc1 = mx.sym.FullyConnected(data, name='fc1', num_hidden=128)
>>> act1 = mx.sym.Activation(fc1, name='relu1', act_type='relu')
>>> fc2 = mx.sym.FullyConnected(act1, name='fc2', num_hidden=10)
>>> out = mx.sym.SoftmaxOutput(fc2, name='softmax')
>>> mod = mx.mod.Module(out) # create a module by given a Symbol
```

2) Bind

```python
>>> mod.bind(data_shapes=nd_iter.provide_data,
            label_shapes=nd_iter.provide_label) # create memory by given input shapes
>>> mod.init_params() # initial parameters with the default random initializer
```

3) Pass data

```python
>>> mod.fit(nd_iter, num_epoch=10, ...) # train
>>> mod.predict(new_nd_iter) # predict on new data
```
Outline

• Review of Deep Learning

• Apache MXNet Framework

• Distributed Inference using MXNet and Spark
Distributed Inference Challenges

- Similar to large scale data processing systems

Apache Spark:

- Multiple Cluster Managers
- Works well with MXNet.
- Integrates with Hadoop & big data tools.

High Performance DL framework

Distributed Cluster

Resource Management

Job Management

Efficient Partition of Data

Deep Learning Setup
MXNet + Spark for Inference.

- For demo, CIFAR-10 test dataset with 10K Images.
- PySpark on Amazon EMR, MXNet is also available in Scala.
- Inference on CPUs, can be extended to use GPUs.
Distributed Inference Pipeline

- download S3 keys on driver
- create RDD and partition
- fetch batch of images on executor
- decode to numpy array
- run prediction
- initialize model only once
- mapPartitions
- collect predictions
MXNet + Spark for Inference.

```python
conf = SparkConf().setAppName("Distributed Inference using MXNet and Spark")
conf.set('spark.executor.cores', '1')
n_partitions = len(keys) // args['batch']

rdd = sc.parallelize(keys, num_slices=n_partitions)
sc.broadcast(args['bucket'])

rdd = rdd.mapPartitions(lambda k: download_objects(args['bucket'], k))
rdd = rdd.mapPartitions(load_images)
sc.broadcast(args)

rdd = rdd.mapPartitions(lambda imgs: predict(imgs, args))
output = rdd.collect()
```
class MXModel(object):
    
    This is a singleton class that just holds the loaded mxnet model in the module object. We don't want to load the model for every inference when called from the map method.
    
    __metaclass__ = Singleton
    model_loaded = False
    mod = None
    synsets = None

    def __init__(self, sym_url, param_url, synset_url, batch_size):
        (s_fname, p_fname, synset_fname) = self.download_model_files(sym_url, param_url, synset_url)
        MXModel.synsets = self.load_synset(synset_fname)
        MXModel.mod = self.init_module(s_fname, p_fname, batch_size)
        MXModel.model_loaded = True

    def predict(img_batch, args):
        
        Run predication on batch of images in 4-D numpy array format and return the top_5 probability along with their classes.

        
        import mxnet as mx
        import numpy as np
        
        logger.info('predict-args:%s' %(args))

        if not MXModel.model_loaded:
            MXModel(args['sym_url'], args['param_url'], args['label_url'], args['batch'])

        MXModel.mod.forward(Batch([mx.nd.array(img_batch)]))
Summary

• Overview of Deep Learning
  o How Deep Learning works and Why Deep Learning is a big deal.
  o Phases of Deep Learning
  o Types of Learning

• Apache MXNet – Efficient deep learning library
  o NDArray/Symbol/Module

• Apache MXNet and Spark for distributed Inference.
What’s Next?

• Released simplified Scala Inference APIs (v1.2.0)

• Working on Java APIs for Inference.

• Dataframe support is under consideration.

• MXNet community is fast evolving, join hands to democratize AI.
Resources/References

- [https://github.com/apache/incubator-mxnet](https://github.com/apache/incubator-mxnet)
- [Blog- Distributed Inference using MXNet and Spark](https://example.com/blog)
- [Distributed Inference code sample on GitHub](https://example.com/distributed)
- [Apache MXNet Gluon Tutorials](https://example.com/gluon)
- [Apache MXNet – Flexible and efficient deep learning.](https://example.com/mxnet)
- [The Deep Learning Book](https://example.com/deep-learning)
- [MXNet – Using pre-trained models](https://example.com/mxnet/pretrained)
- [Amazon Elastic MapReduce](https://example.com/elasticmapreduce)
Thank You
nswamy@apache.org