Intro to Deep Learning Frameworks

TensorFlow

TensorFlow Serving

TensorFlow Lite

TensorFlow

- TensorFlow is an open source software library for numerical computation using data flow graphs.
- Supports both large scale training and inference.
- Map the nodes of a dataflow graph across
  - many machines in a cluster
  - within a machine across multiple computational devices(CPUs, GPUs, TPUs)
- Deep Flexibility

TensorFlow
Programming model

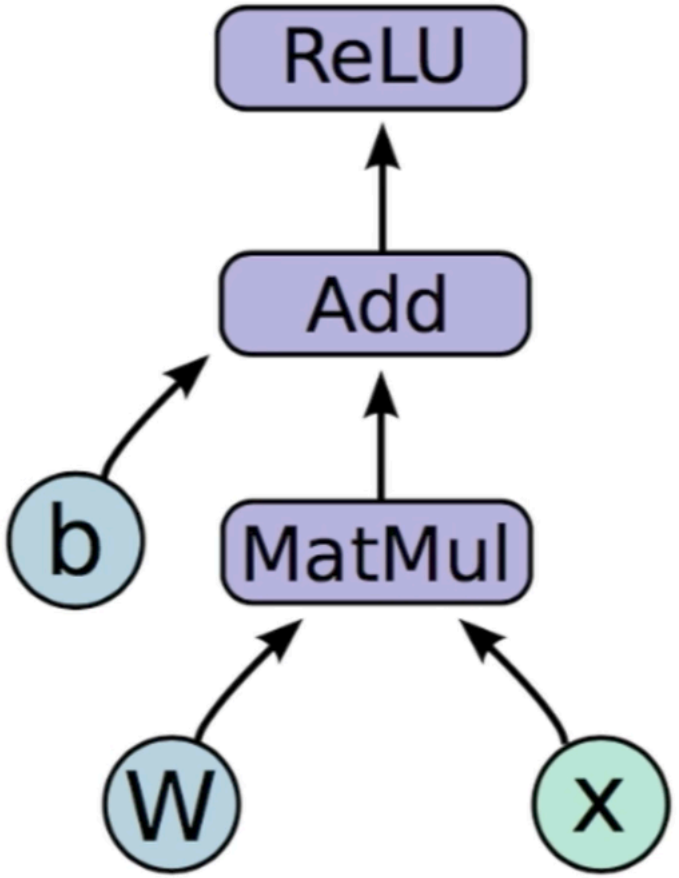
Big idea: express a numerical computation as a graph

- Graph **nodes** are **operations** which have any number of inputs and outputs.
- Graph **edges** are **tensors** (multi-dimensional arrays) which flow between nodes.

TensorFlow
Programming model

Y = ReLU(W * x + b)

1. Building the computational graph.
2. Running the computational graph.



TensorFlow
Programming model

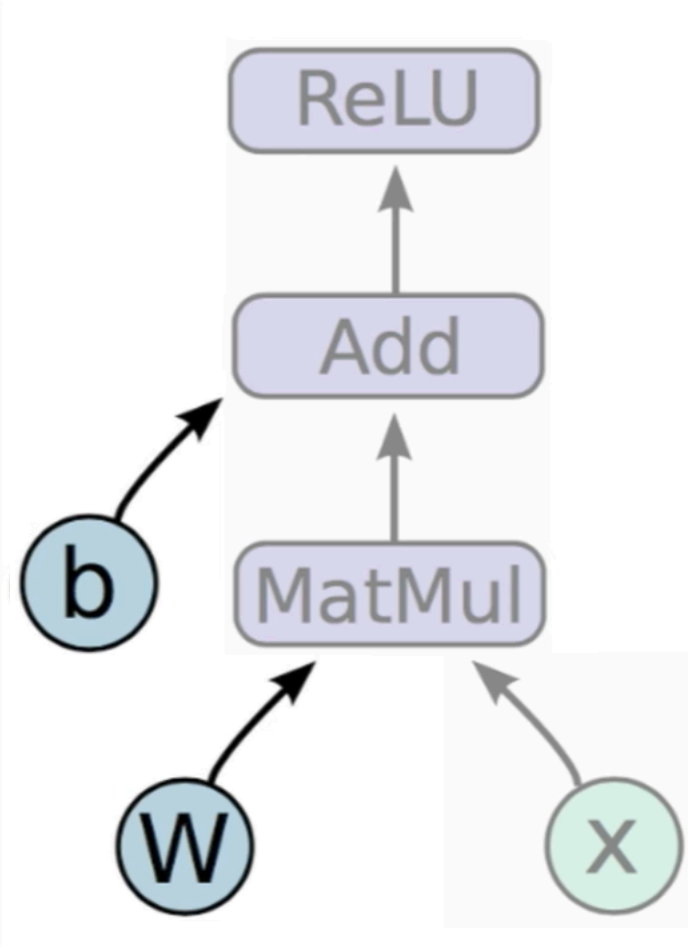
Y = ReLU(Wx + b)

Variables :

Stateful nodes which output their current value .

State is retained across multiple executions of a graph.

(mostly parameters)



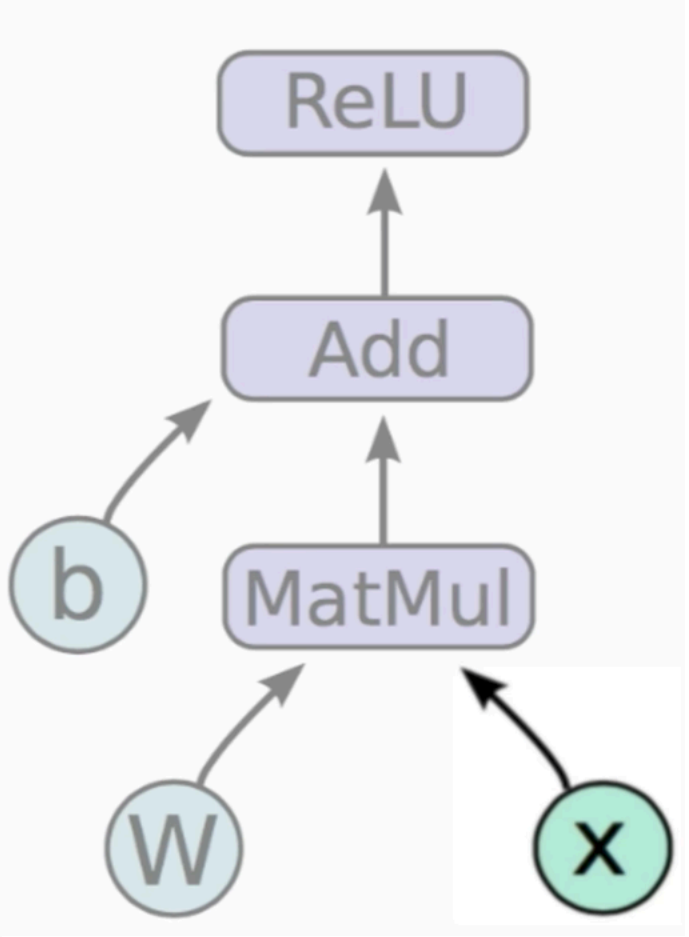
TensorFlow
Programming model

Y = ReLU(Wx + b)

Placeholders:

Nodes whose value is fed in at execution time.

(inputs, labels, …)



TensorFlow
Programming model

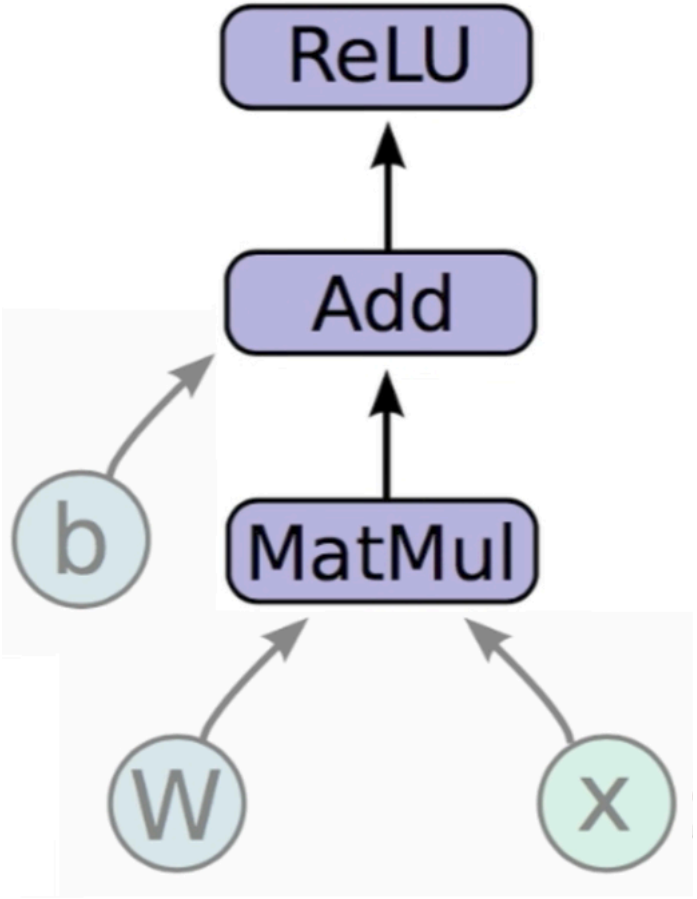
Y = ReLU(Wx + b)

Mathematical operations：

MatMul： Multi two matrix values.

Add: Add elementwise.

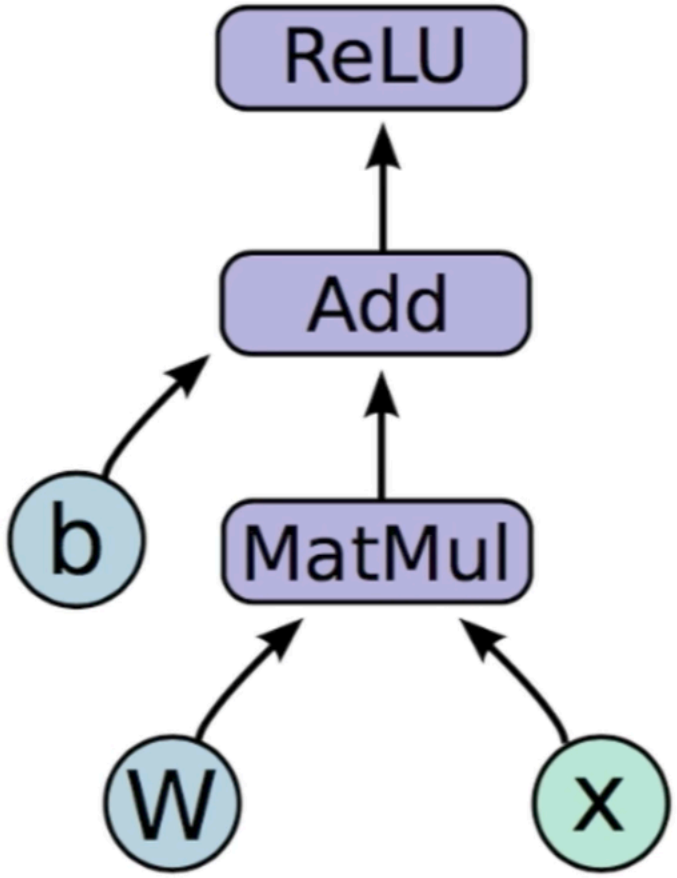
ReLU: Activate with elementwise rectified linear function.



TensorFlow
Programming model

Y = ReLU(Wx + b)

```
import tensorflow as tf

b = tf.Variable(tf.zeros((100, )))
W = tf.Variable(tf.random_uniform((784, 100), -1, 1))

x = tf.placeholder(tf.float32, (100, 784))

y = tf.nn.relu(tf.matmul(x, W) + b)
```



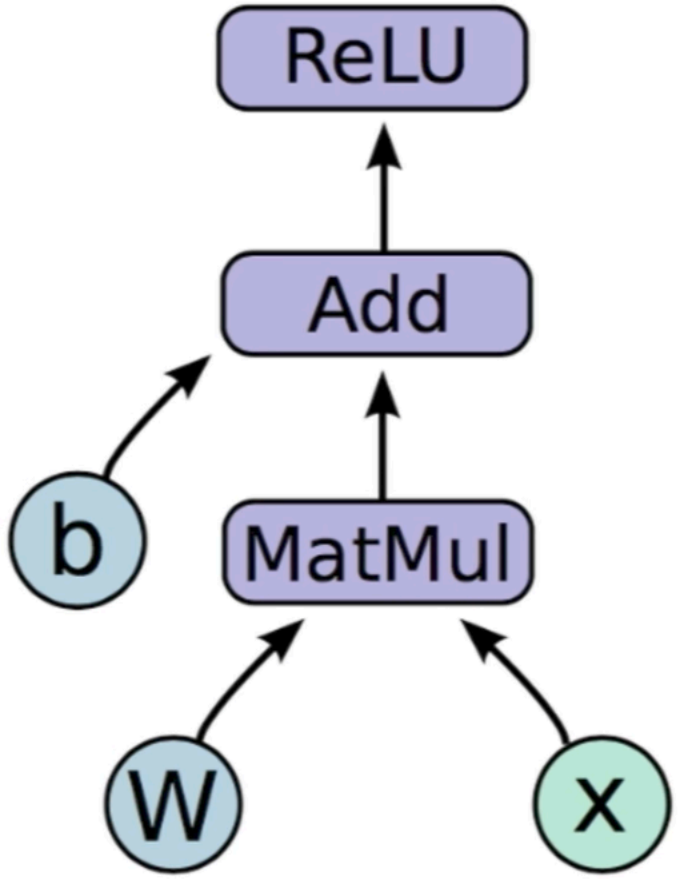
TensorFlow

How do we run it?

So far we have defined a graph.

We can deploy this graph with a **Session**:

A binding to a particular execution context(e.g. CPU, GPU)



TensorFlow
Getting output

sess.run(fetches, feeds)

Fetches: list of graph nodes. Return the outputs of these nodes.

Feeds: Dictionary mapping from graph to concrete values. Specifies the value of each graph node given in the dictionary.

```
import tensorflow as tf
import numpy as np

b = tf.Variable(tf.zeros((100, )))
W = tf.Variable(tf.random_uniform((784, 100), -1, 1))

x = tf.placeholder(tf.float32, (100, 784))

y = tf.nn.relu(tf.matmul(x, W) + b)

sess = tf.Session()
sess.run(tf.initialize_all_variables())
sess.run(y, {x: np.random.random(100, 784)})
```

TensorFlow

Sample: MNIST

MNIST is a classic problem in machine learning. The problem is to look at greyscale 28x28 pixel images of handwritten digits and determine which digit the image represents, for all the digits from zero to nine.

TensorFlow
Sample: MNIST

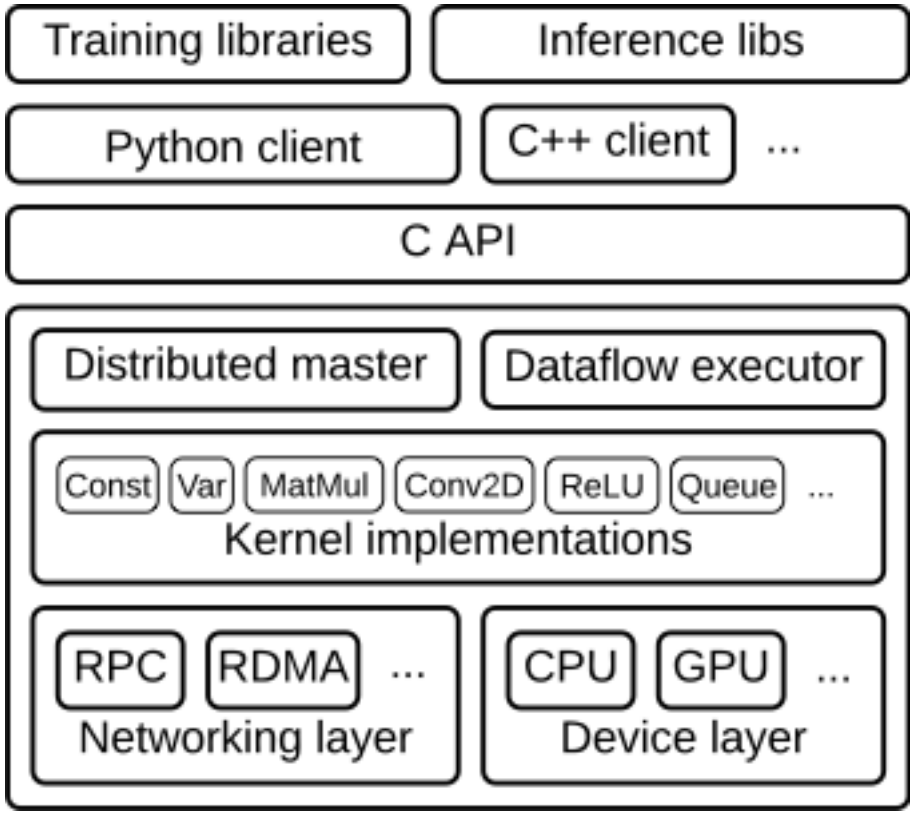
```
from __future__ import absolute_import
from __future__ import division
from __future__ import print_function

import gzip
import os
import tempfile

import numpy
from six.moves import urllib
from six.moves import xrange  # pylint: disable=redefined-builtin
import tensorflow as tf

import mnist_input_data
# from tensorflow.contrib.learn.python.learn.datasets.mnist import read_data_sets
"""""""


# import tensorflow.examples.tutorials.mnist.input_data
# mnist = input_data.read_data_sets("MNIST_data/", one_hot=True)
mnist = mnist_input_data.read_data_sets(".", one_hot=True)

x = tf.placeholder(tf.float32, [None, 784])
#"initialize weight W and bias b"
W = tf.Variable(tf.zeros([784, 10]))
b = tf.Variable(tf.zeros([10]))
```

downloading and read data

define a graph

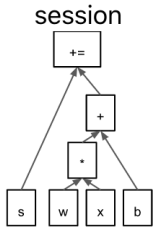
TensorFlow
Architecture

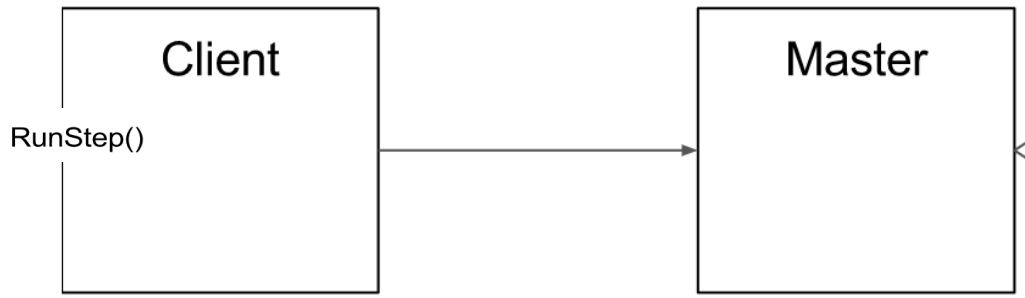
- Client:
- Distributed Master
- Worker Service (one for each task)
- Kernel Implementation



https://www.tensorflow.org/extend/architecture

TensorFlow
Architecture





Client:

•Defines the computation as a dataflow graph.

•Initiates graph execution using a session

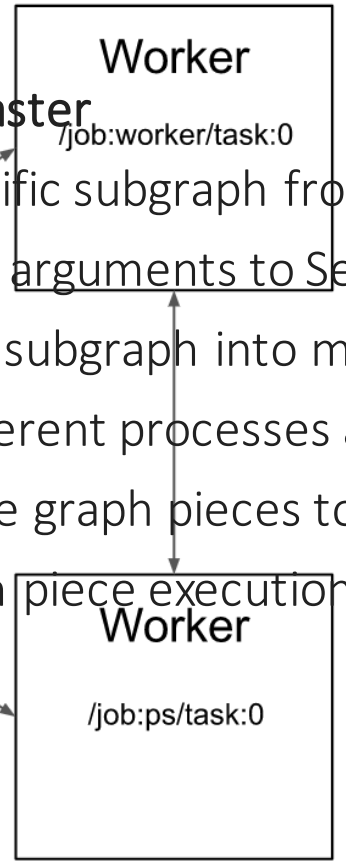
Distributed Master

•Prunes a specific subgraph from the graph, as defined by the arguments to Session.run().

•Partitions the subgraph into multiple pieces that run in different processes and devices.

•Distributes the graph pieces to worker services.

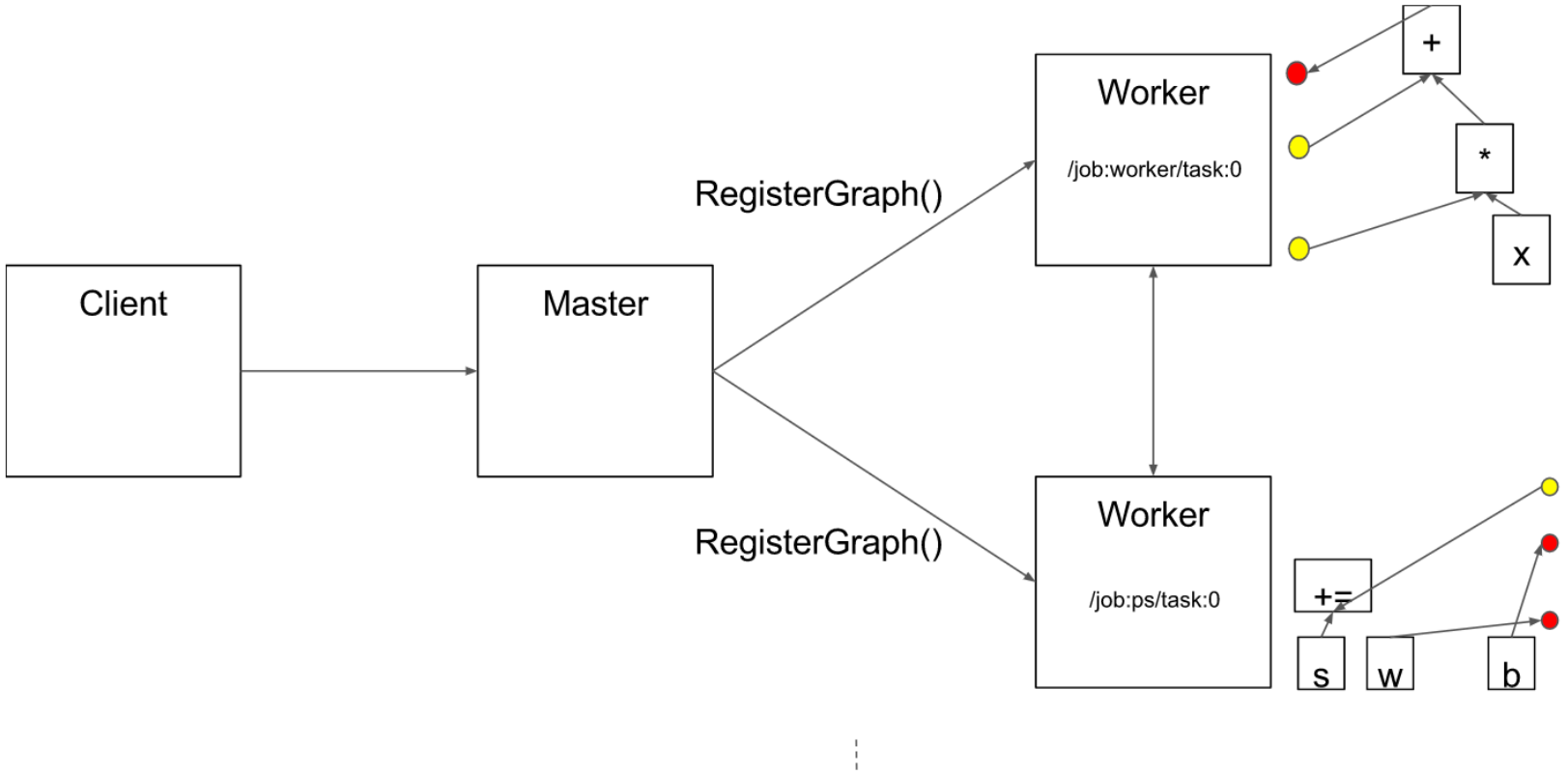
•Initiates graph piece execution by worker services.



vice only

TensorFlow

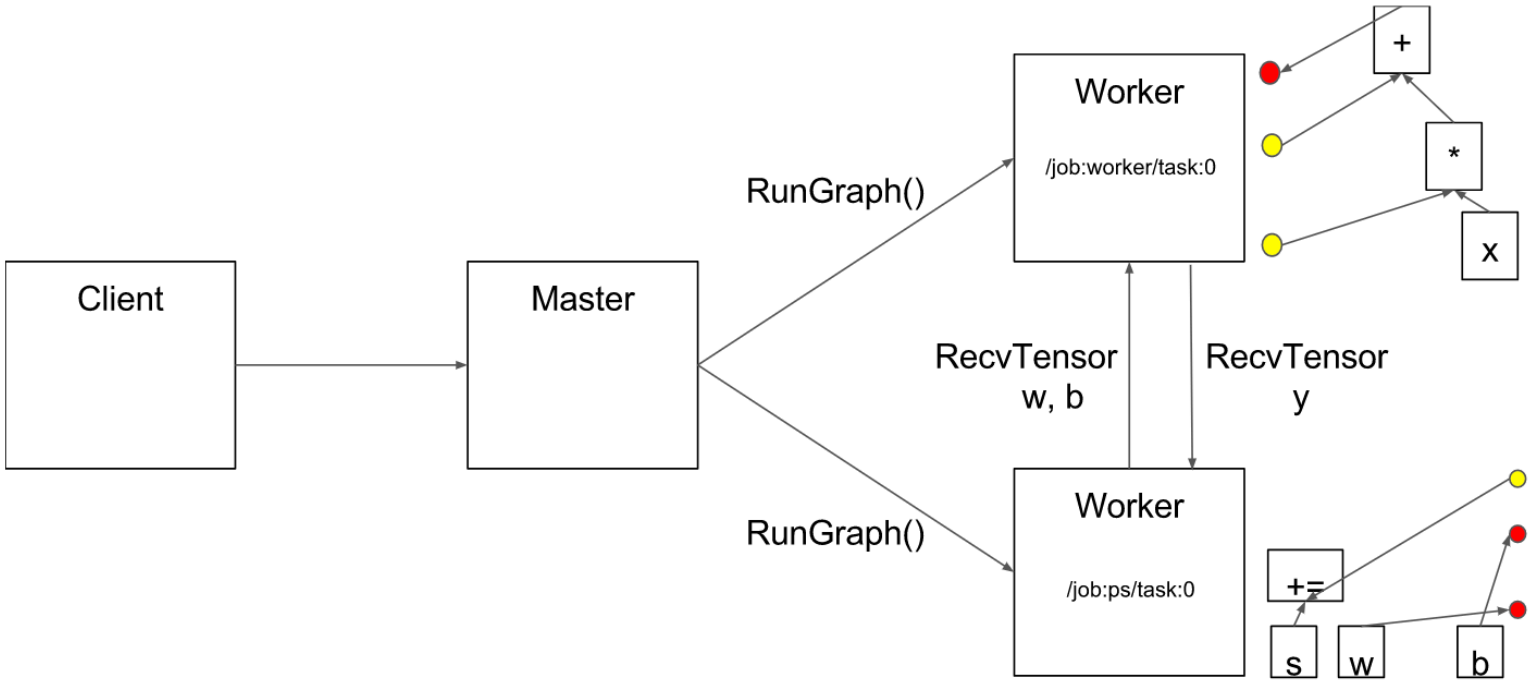
Architecture a possible partition of our example graph



TensorFlow
Architecture

Worker Service：

- handles requests from the master;
- schedules the execution of the kernels for the operations that comprise a local subgraph, and mediates direct communication between tasks.



gRPC over TCP,
RDMA over Converged Ethernet,
...

Distributed TensorFlow

One machine with single GPU

```
#coding=utf-8
#单机单卡
#对于单机单卡，可以把参数和计算都定义再gpu上，不过如果参数模型比较大，显存不足等情况，就得放在cpu上
import  tensorflow as tf

with tf.device('/cpu:0'):#也可以放在gpu上
    w=tf.get_variable('w',(2,2),tf.float32,initializer=tf.constant_initializer(2))
    b=tf.get_variable('b',(2,2),tf.float32,initializer=tf.constant_initializer(5))

with tf.device('/gpu:0'):
    addwb=w+b
    mutwb=w*b


ini=tf.initialize_all_variables()
with tf.Session() as sess:
    sess.run(ini)
    np1,np2=sess.run([addwb,mutwb])
    print np1
    print np2
```

Distributed TensorFlow

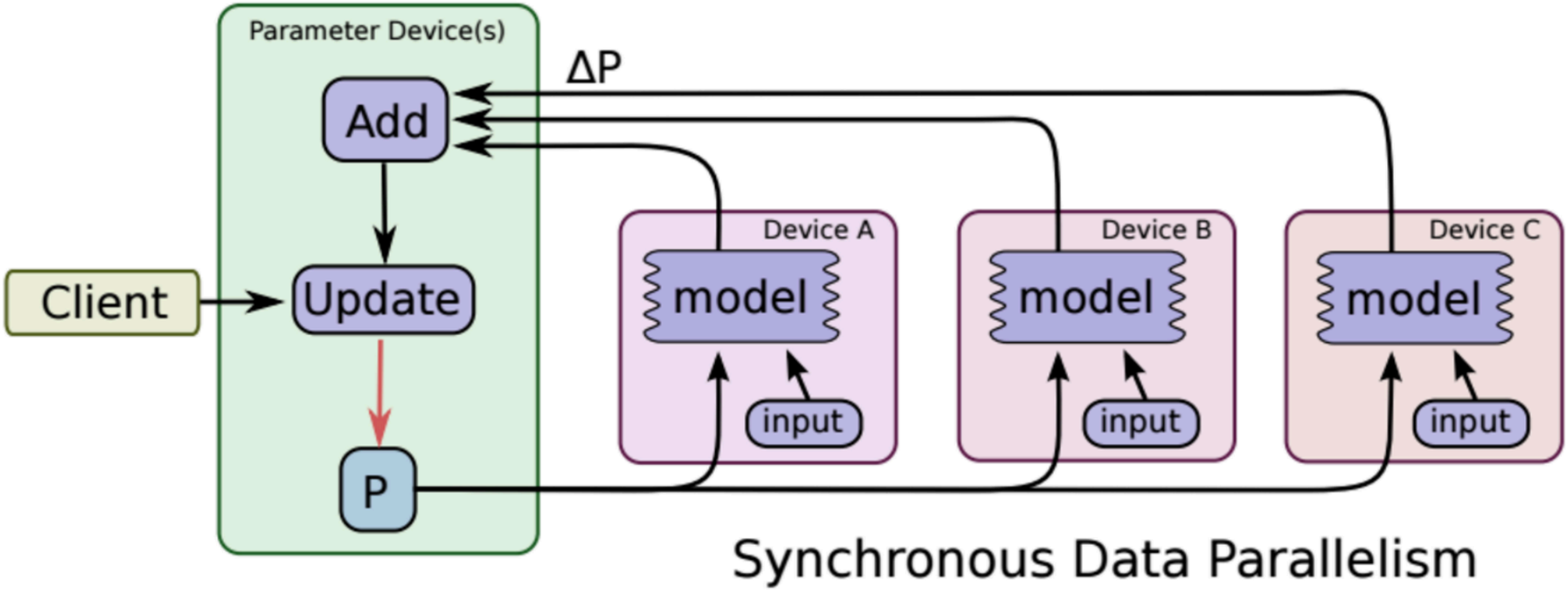
One machine with multi-GPU


```
import  tensorflow as tf

with tf.device('/cpu:0'):
    w=tf.get_variable('w',(2,2),tf.float32,initializer=tf.constant_initializer(2))
    b=tf.get_variable('b',(2,2),tf.float32,initializer=tf.constant_initializer(5))

with tf.device('/gpu:0'):
    addwb=w+b
with tf.device('/gpu:1'):
    mutwb=w*b


ini=tf.initialize_all_variables()
with tf.Session() as sess:
    sess.run(ini)
    while 1:
        print sess.run([addwb,mutwb])
```

http://blog.csdn.net/hjimce/article/details/61197190

Distributed TensorFlow

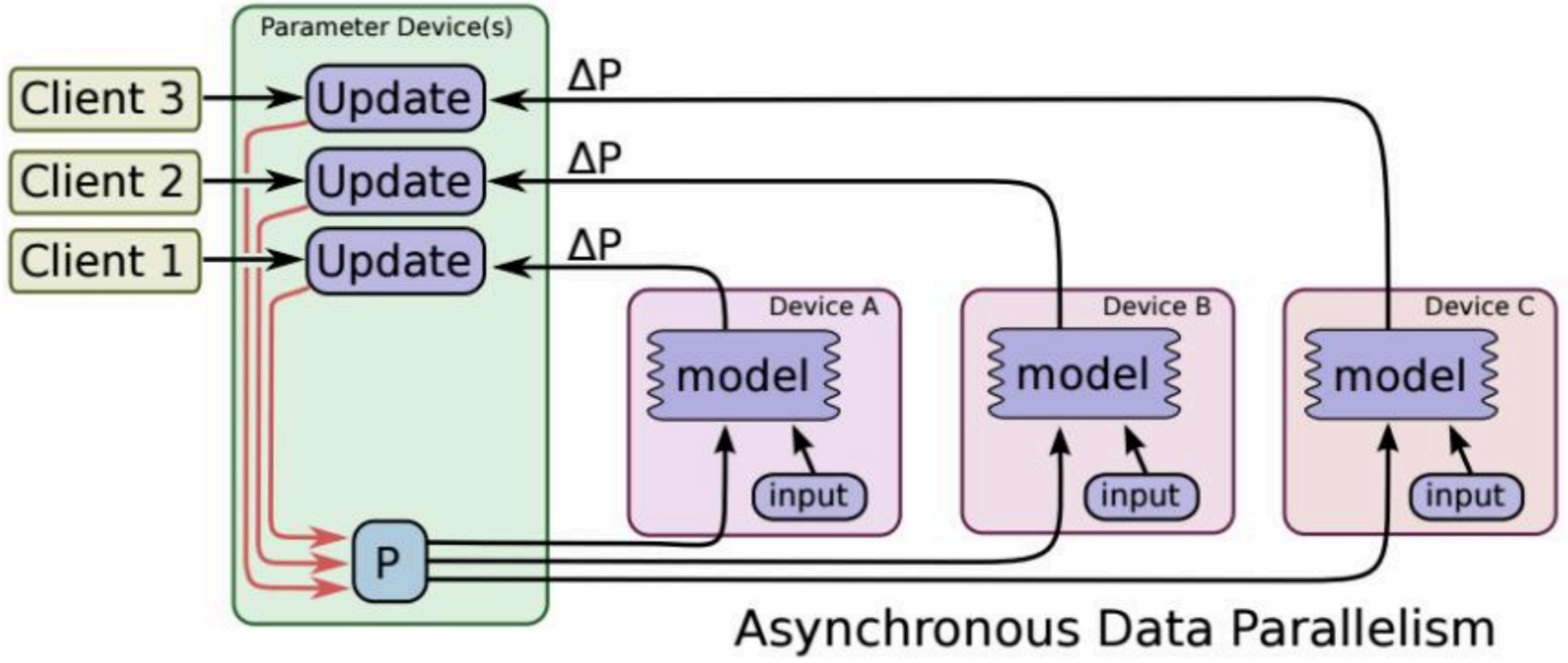
Multiple machines with multi-GPU



http://blog.csdn.net/hjimce/article/details/61197190

Distributed TensorFlow

Multiple machines with multi-GPU



http://blog.csdn.net/hjimce/article/details/61197190

TensorFlow Cluster
Implementation

1. Define the cluster.
2. Define a server for each device.
3. Run different code on each device

```
# suppose our cluster has four machines A, B, C, D,
import  tensorflow as tf

cluster=tf.train.ClusterSpec({
    "worker": [
        "A_IP:2222",
        "B_IP:1234"
        "C_IP:2222"
    ],
    "ps": [
        "D_IP:2222",
    ]})
```

```
# For each device, we should define a server.
server=tf.train.Server(cluster,job_name='worker',task_index=0)
```

```
with tf.device('/job:ps/task:0'):    # device D
    w=tf.get_variable('w',(2,2),tf.float32,initializer=tf.constant_initializer(2))
    b=tf.get_variable('b',(2,2),tf.float32,initializer=tf.constant_initializer(5))

with tf.device('/job:worker/task:0/cpu:0'): #device A
    addwb=w+b
with tf.device('/job:worker/task:1/cpu:0'): #device B
    mutwb=w*b
with tf.device('/job:worker/task:2/cpu:0'): #device C
    divwb=w/b
```

TensorFlow
Supported devices

- On a system with GPUs, TensorFlow can **automatically** determine which devices each operation will be assigned to.
- If a TensorFlow operation has both CPU and GPU implementations, the GPU devices will be given priority when the operation is assigned to a device.
- Allow **manually** specifying the device for each operation in a system with GPUs. In TensorFlow, the supported device types are CPU and GPU. They are represented as strings. For example:
  - "/cpu:0": The CPU of your machine.
  - "/device:GPU:0": The GPU of your machine, if you have one.
  - “/device:GPU:1”: The second GPU of your machine, etc.

TensorFlow
Logging Device placement

```
# Creates a graph.
a = tf.constant([1.0, 2.0, 3.0, 4.0, 5.0, 6.0], shape=[2, 3], name='a')
b = tf.constant([1.0, 2.0, 3.0, 4.0, 5.0, 6.0], shape=[3, 2], name='b')
c = tf.matmul(a, b)
# Creates a session with log_device_placement set to True.
sess = tf.Session(config=tf.ConfigProto(log_device_placement=True))
# Runs the op.
print(sess.run(c))
```

```
Device mapping:
/job:localhost/replica:0/task:0/device:GPU:0 -> device: 0, name: Tesla K40c, pci bus
id: 0000:05:00.0
b: /job:localhost/replica:0/task:0/device:GPU:0
a: /job:localhost/replica:0/task:0/device:GPU:0
MatMul: /job:localhost/replica:0/task:0/device:GPU:0
[[ 22.  28.]
 [ 49.  64.]]
```

TensorFlow
Manual device placement

```
# Creates a graph.
with tf.device('/cpu:0'):
  a = tf.constant([1.0, 2.0, 3.0, 4.0, 5.0, 6.0], shape=[2, 3], name='a')
  b = tf.constant([1.0, 2.0, 3.0, 4.0, 5.0, 6.0], shape=[3, 2], name='b')
c = tf.matmul(a, b)
# Creates a session with log_device_placement set to True.
sess = tf.Session(config=tf.ConfigProto(log_device_placement=True))
# Runs the op.
print(sess.run(c))
```

```
Device mapping:
/job:localhost/replica:0/task:0/device:GPU:0 -> device: 0, name: Tesla K40c, pci bus
id: 0000:05:00.0
b: /job:localhost/replica:0/task:0/cpu:0
a: /job:localhost/replica:0/task:0/cpu:0
MatMul: /job:localhost/replica:0/task:0/device:GPU:0
[[ 22.  28.]
 [ 49.  64.]]
```

TensorFlow
Cluster

- “**Note:** Manually specifying these cluster specifications can be tedious, especially for large clusters. We are working on tools for launching tasks programmatically, e.g. using a cluster manager like Kubernetes. If there are particular cluster managers for which you‘d like to see support, please raise a GitHub issue.”

https://www.tensorflow.org/deploy/distributed

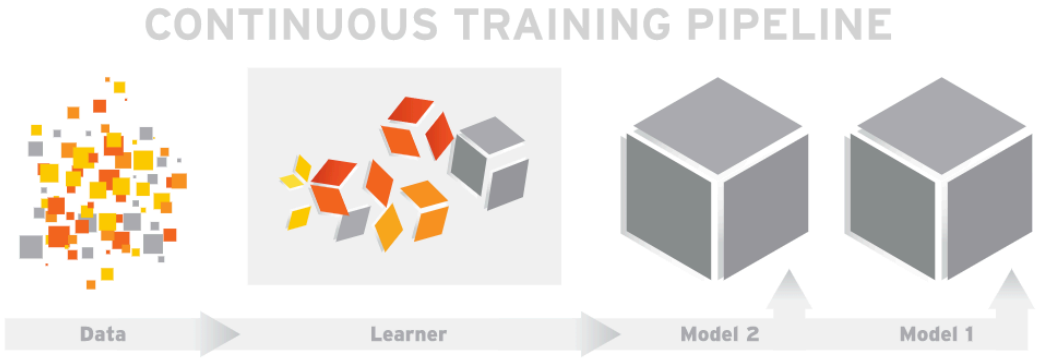
TensorFlow

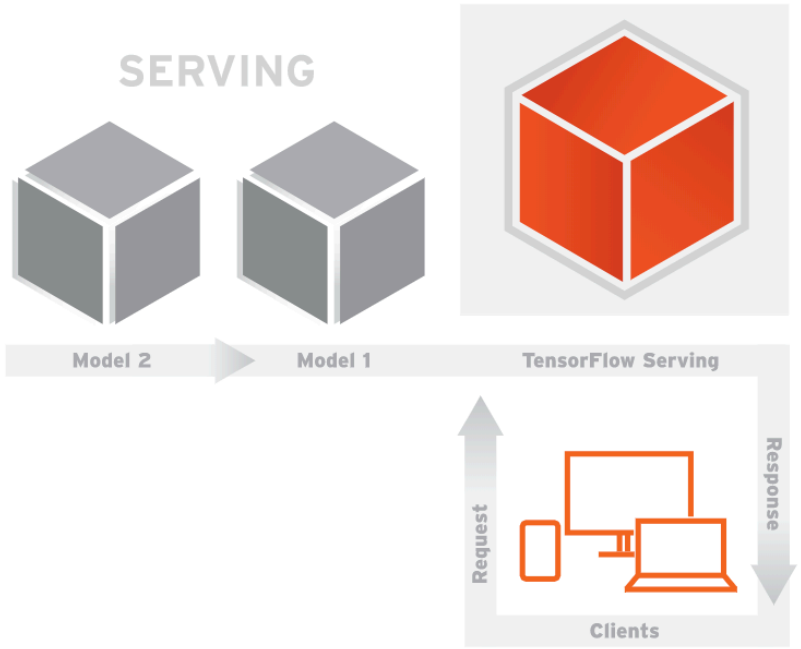
Programming Language – R vs Python

- ”TensorFlow supports multiple client languages, and we have prioritized Python and C++, because our internal users are most familiar with these languages”
- Most of the training libraries are still Python-only, but C++ does have support for efficient inference.
- “The balance now shifted towards Python as it had an enormous list of Deep Learning libraries and frameworks which R lacked .”
- “Python was now leading the Deep Learning World in every way it can, because with R it was almost impossible to run big complex deep learning Models , but not anymore, R is back in the fight again .”

TensorFlow Serving

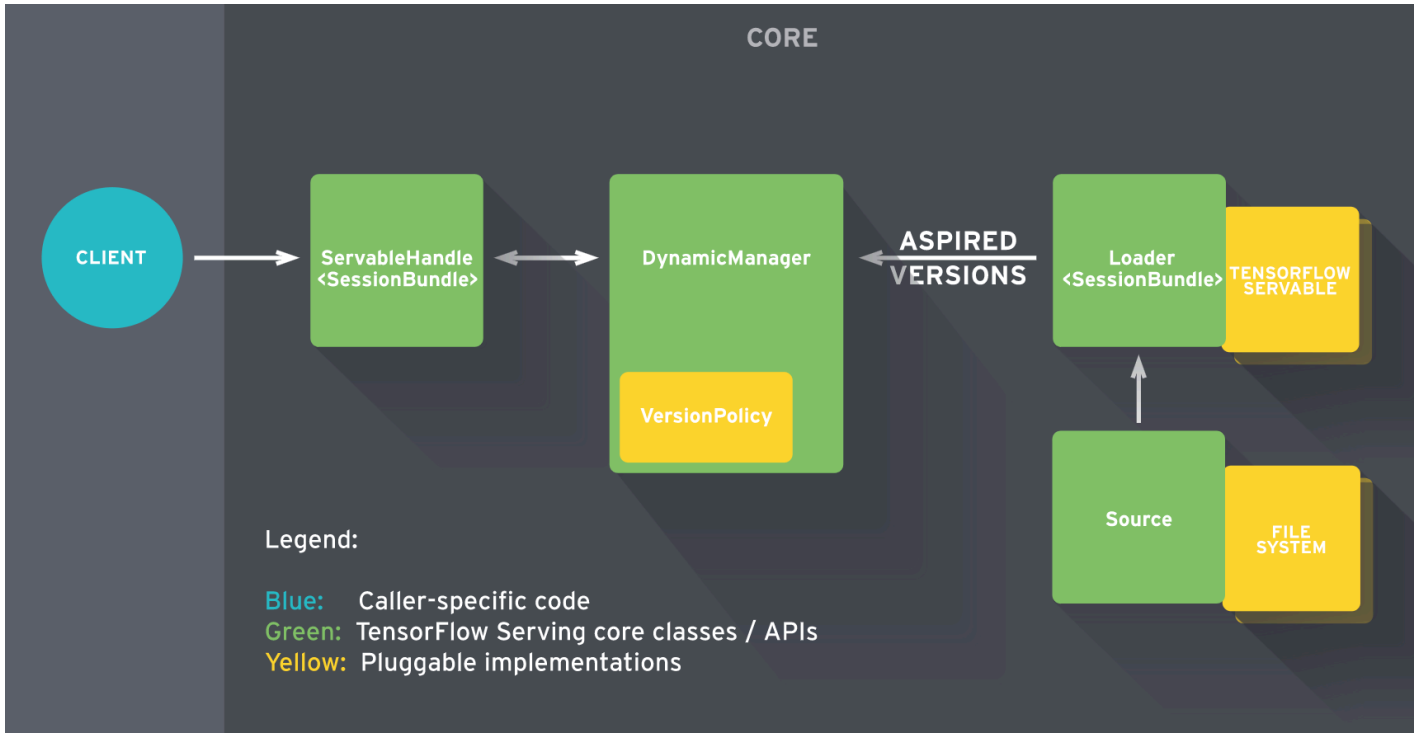
- Deal with the **inference** aspect of machine learning.
- Take models after training.
- Manage models' lifetimes.
- **Version transitions.**





https://www.tensorflow.org/serving/

TensorFlow Serving
Architecture



Sources

- plugin modules that originate servables;
- each Source originates zero or more servable streams.
- a Source supplies one Loader instance for each version it wants to have loaded.

Servables:

TensorFlow Serving represents a model as one or more servables, including:

- loading Servables
- serving Servables
- unloading Servables
- Version transitions

Loaders

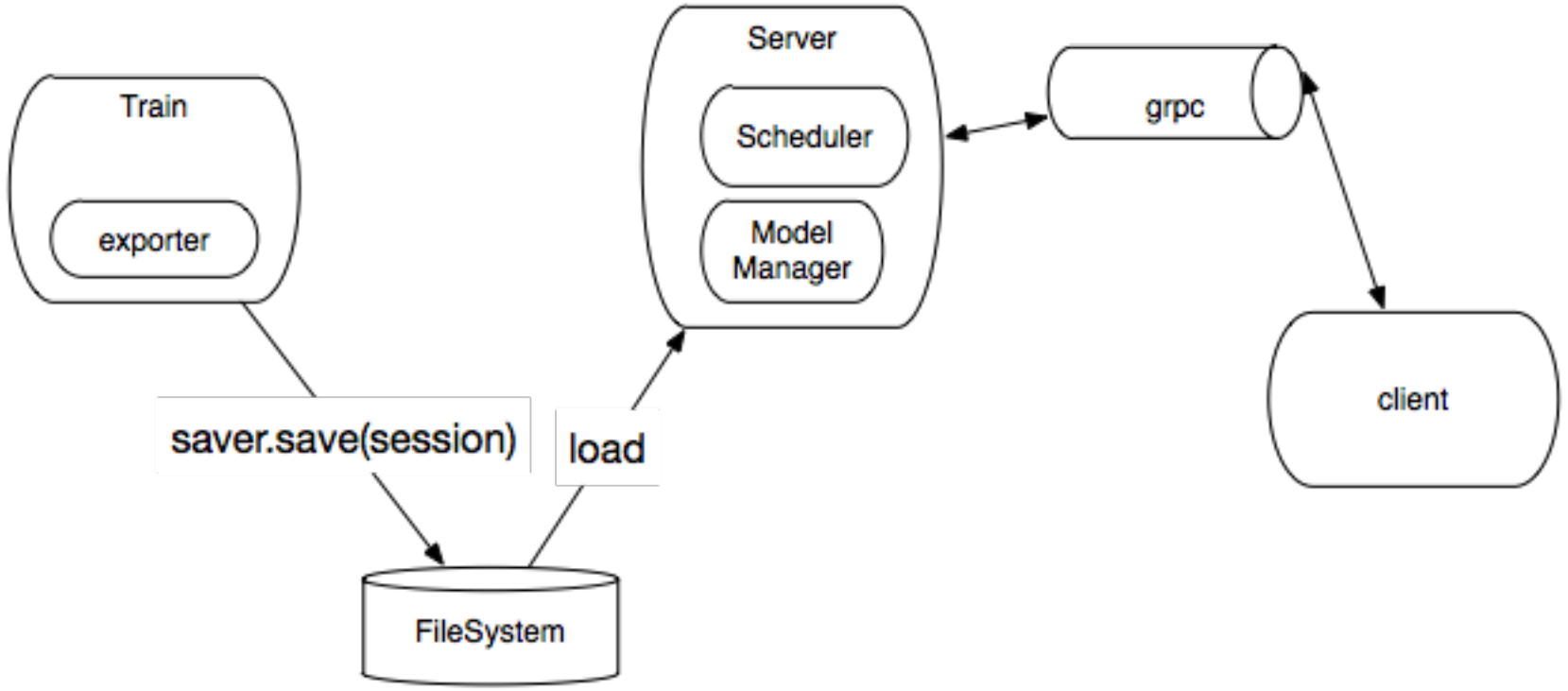
- manage a servable's life cycle.
- Specifically, Loaders standardize the APIs for loading and unloading a servable

Aspired versions

represent the set of servable versions that should be loaded and ready

https://www.tensorflow.org/serving/architecture_overview

TensorFlow Serving
Workflow



Tenso
Servin

1. The t

```
# Train model
print 'Training model...'
mnist = mnist_input_data.read_data_sets(FLAGS.work_dir, one_hot=True)

sess = tf.InteractiveSession()
serialized_tf_example = tf.placeholder(tf.string, name='tf_example')
feature_configs = {'x': tf.FixedLenFeature(shape=[784], dtype=tf.float32),}
tf_example = tf.parse_example(serialized_tf_example, feature_configs)
x = tf.identity(tf_example['x'], name='x')  # use tf.identity() to assign name
y_ = tf.placeholder('float', shape=[None, 10])

w = tf.Variable(tf.zeros([784, 10]))
b = tf.Variable(tf.zeros([10]))

sess.run(tf.global_variables_initializer())
y = tf.nn.softmax(tf.matmul(x, w) + b, name='y')
cross_entropy = -tf.reduce_sum(y_ * tf.log(y))
train_step = tf.train.GradientDescentOptimizer(0.01).minimize(cross_entropy)
values, indices = tf.nn.top_k(y, 10)
table = tf.contrib.lookup.index_to_string_table_from_tensor(
    tf.constant([str(i) for i in xrange(10)]))
prediction_classes = table.lookup(tf.to_int64(indices))

for _ in range(FLAGS.training_iteration):
  batch = mnist.train.next_batch(50)
  train_step.run(feed_dict={x: batch[0], y_: batch[1]})

correct_prediction = tf.equal(tf.argmax(y, 1), tf.argmax(y_, 1))
accuracy = tf.reduce_mean(tf.cast(correct_prediction, 'float'))
print 'training accuracy %g' % sess.run(
    accuracy, feed_dict={x: mnist.test.images,
                         y_: mnist.test.labels})
print 'Done training!'
```

TensorFlow Serving

Serving in a TensorFlow Model

2. Use TensorFlow's **SavedModelBuilder** module to export the model.

```
# Export model
# WARNING(break-tutorial-inline-code): The following code snippet is
# in-lined in tutorials, please update tutorial documents accordingly
# whenever code changes.
export_path_base = sys.argv[-1]
export_path = os.path.join(
    tf.compat.as_bytes(export_path_base),
    tf.compat.as_bytes(str(FLAGS.model_version)))
print 'Exporting trained model to', export_path
builder = tf.saved_model.builder.SavedModelBuilder(export_path)
```

```
if len(sys.argv) < 2 or sys.argv[-1].startswith('-'):
  print('Usage: mnist_export.py [--training_iteration=x] '
        '[--model_version=y] export_dir')
  sys.exit(-1)
```

Tens
Servi

2. Use T el.

```
# Build the signature_def_map.
classification_inputs = tf.saved_model.utils.build_tensor_info(
    serialized_tf_example)
classification_outputs_classes = tf.saved_model.utils.build_tensor_info(
    prediction_classes)
classification_outputs_scores = tf.saved_model.utils.build_tensor_info(values)

classification_signature = (
    tf.saved_model.signature_def_utils.build_signature_def(
        inputs={
            tf.saved_model.signature_constants.CLASSIFY_INPUTS:
                classification_inputs
        },
        outputs={
            tf.saved_model.signature_constants.CLASSIFY_OUTPUT_CLASSES:
                classification_outputs_classes,
            tf.saved_model.signature_constants.CLASSIFY_OUTPUT_SCORES:
                classification_outputs_scores
        },
        method_name=tf.saved_model.signature_constants.CLASSIFY_METHOD_NAME))

tensor_info_x = tf.saved_model.utils.build_tensor_info(x)
tensor_info_y = tf.saved_model.utils.build_tensor_info(y)

prediction_signature = (
    tf.saved_model.signature_def_utils.build_signature_def(
        inputs={'images': tensor_info_x},
        outputs={'scores': tensor_info_y},
        method_name=tf.saved_model.signature_constants.PREDICT_METHOD_NAME))

legacy_init_op = tf.group(tf.tables_initializer(), name='legacy_init_op')
builder.add_meta_graph_and_variables(
    sess, [tf.saved_model.tag_constants.SERVING],
    signature_def_map={
        'predict_images':
            prediction_signature,
        tf.saved_model.signature_constants.DEFAULT_SERVING_SIGNATURE_DEF_KEY:
            classification_signature,
    },
    legacy_init_op=legacy_init_op)

builder.save()

print 'Done exporting!'
```

TensorFlow Serving

Serving in a TensorFlow Model

```
net@net-r4:~/0/serving/tensorflow_serving/example$ python mnist_saved_model.py --model_version=1 /tmp/mnist_model
Training model...
Extracting /tmp/train-images-idx3-ubyte.gz
Extracting /tmp/train-labels-idx1-ubyte.gz
Extracting /tmp/t10k-images-idx3-ubyte.gz
Extracting /tmp/t10k-labels-idx1-ubyte.gz
2017-12-26 14:05:34.038660: I tensorflow/core/platform/cpu_feature_guard.cc:137] Your CPU supports instructions that this TensorFlow binary was not compiled to use: SSE4.1 SSE4.2
 AVX AVX2 FMA
2017-12-26 14:05:34.251896: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1030] Found device 0 with properties:
name: GeForce GTX 1080 major: 6 minor: 1 memoryClockRate(GHz): 1.835
pciBusID: 0000:02:00.0
totalMemory: 7.92GiB freeMemory: 7.77GiB
2017-12-26 14:05:34.440881: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1030] Found device 1 with properties:
name: GeForce GTX 1080 major: 6 minor: 1 memoryClockRate(GHz): 1.835
pciBusID: 0000:04:00.0
totalMemory: 7.92GiB freeMemory: 7.80GiB
2017-12-26 14:05:34.441332: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1045] Device peer to peer matrix
2017-12-26 14:05:34.442986: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1051] DMA: 0 1
2017-12-26 14:05:34.442998: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1061] 0:   Y Y
2017-12-26 14:05:34.443002: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1061] 1:   Y Y
2017-12-26 14:05:34.443009: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1120] Creating TensorFlow device (/device:GPU:0) -> (device: 0, name: GeForce GTX 1080, pci bus id:
 0000:02:00.0, compute capability: 6.1)
2017-12-26 14:05:34.443014: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1120] Creating TensorFlow device (/device:GPU:1) -> (device: 1, name: GeForce GTX 1080, pci bus id:
 0000:04:00.0, compute capability: 6.1)
training accuracy 0.9092
Done training!
Exporting trained model to /tmp/mnist_model/1
Done exporting!
```

download and read data for training

detect availabel device

done training

done exporting

TensorFlow Serving
Serving in a TensorFlow Model

```
net@net-r4:~/0/serving/tensorflow_serving/example$ ls /tmp/mnist_model/
1
net@net-r4:~/0/serving/tensorflow_serving/example$ ls /tmp/mnist_model/1
saved_model.pb  variables
```

Each version sub-directory contains the following files:

- **saved_model.pb** is the serialized tensorflow::SavedModel. It includes one or more graph definitions of the model, as well as metadata of the model such as signatures.
- **variables** are files that hold the serialized variables of the graphs.

TensorFlow Serving

Load Exported Model With Standard TensorFlow ModelServer

- Then run the server with the following command:

```
net@net-r4:~/0/serving$ tensorflow_model_server --port=9000 --model_name=mnist --model_base_path=/tmp/mnist_model/
2017-12-26 16:46:40.345223: I tensorflow_serving/model_servers/main.cc:147] Building single TensorFlow model file config:  model_name: mnist model_base_path: /tmp/mnist_model/
2017-12-26 16:46:40.345429: I tensorflow_serving/model_servers/server_core.cc:441] Adding/updating models.
2017-12-26 16:46:40.345464: I tensorflow_serving/model_servers/server_core.cc:492]  (Re-)adding model: mnist
2017-12-26 16:46:40.445851: I tensorflow_serving/core/basic_manager.cc:705] Successfully reserved resources to load servable {name: mnist version: 1}
2017-12-26 16:46:40.445887: I tensorflow_serving/core/loader_harness.cc:66] Approving load for servable version {name: mnist version: 1}
2017-12-26 16:46:40.445904: I tensorflow_serving/core/loader_harness.cc:74] Loading servable version {name: mnist version: 1}
2017-12-26 16:46:40.445934: I external/org_tensorflow/tensorflow/contrib/session_bundle/bundle_shim.cc:360] Attempting to load native SavedModelBundle in bundle-shim from: /tmp/m
nist_model/1
2017-12-26 16:46:40.445966: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:236] Loading SavedModel from: /tmp/mnist_model/1
2017-12-26 16:46:40.447744: I external/org_tensorflow/tensorflow/core/platform/cpu_feature_guard.cc:137] Your CPU supports instructions that this TensorFlow binary was not compil
ed to use: AVX2 FMA
2017-12-26 16:46:40.482343: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:155] Restoring SavedModel bundle.
2017-12-26 16:46:40.486518: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:190] Running LegacyInitOp on SavedModel bundle.
2017-12-26 16:46:40.489666: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:284] Loading SavedModel: success. Took 43689 microseconds.
2017-12-26 16:46:40.489942: I tensorflow_serving/core/loader_harness.cc:86] Successfully loaded servable version {name: mnist version: 1}
E1226 16:46:40.492883438    4998 ev_epoll1_linux.c:1051]     grpc epoll fd: 3
2017-12-26 16:46:40.494867: I tensorflow_serving/model_servers/main.cc:288] Running ModelServer at 0.0.0.0:9000 ...
```

```python
def do_inference(hostport, work_dir, concurrency, num_tests):
  """Tests PredictionService with concurrent requests.

  Args:
    hostport: Host:port address of the PredictionService.
    work_dir: The full path of working directory for test data set.
    concurrency: Maximum number of concurrent requests.
    num_tests: Number of test images to use.

  Returns:
    The classification error rate.

  Raises:
    IOError: An error occurred processing test data set.
  """
  test_data_set = mnist_input_data.read_data_sets(work_dir).test
  host, port = hostport.split(':')
  channel = implementations.insecure_channel(host, int(port))
  stub = prediction_service_pb2.beta_create_PredictionService_stub(channel)
  result_counter = _ResultCounter(num_tests, concurrency)
  for _ in range(num_tests):
    request = predict_pb2.PredictRequest()
    request.model_spec.name = 'mnist'
    request.model_spec.signature_name = 'predict_images'
    image, label = test_data_set.next_batch(1)
    request.inputs['images'].CopyFrom(
        tf.contrib.util.make_tensor_proto(image[0], shape=[1, image[0].size]))
    result_counter.throttle()
    result_future = stub.Predict.future(request, 5.0)  # 5 seconds
    result_future.add_done_callback(
        _create_rpc_callback(label[0], result_counter))
  return result_counter.get_error_rate()
```

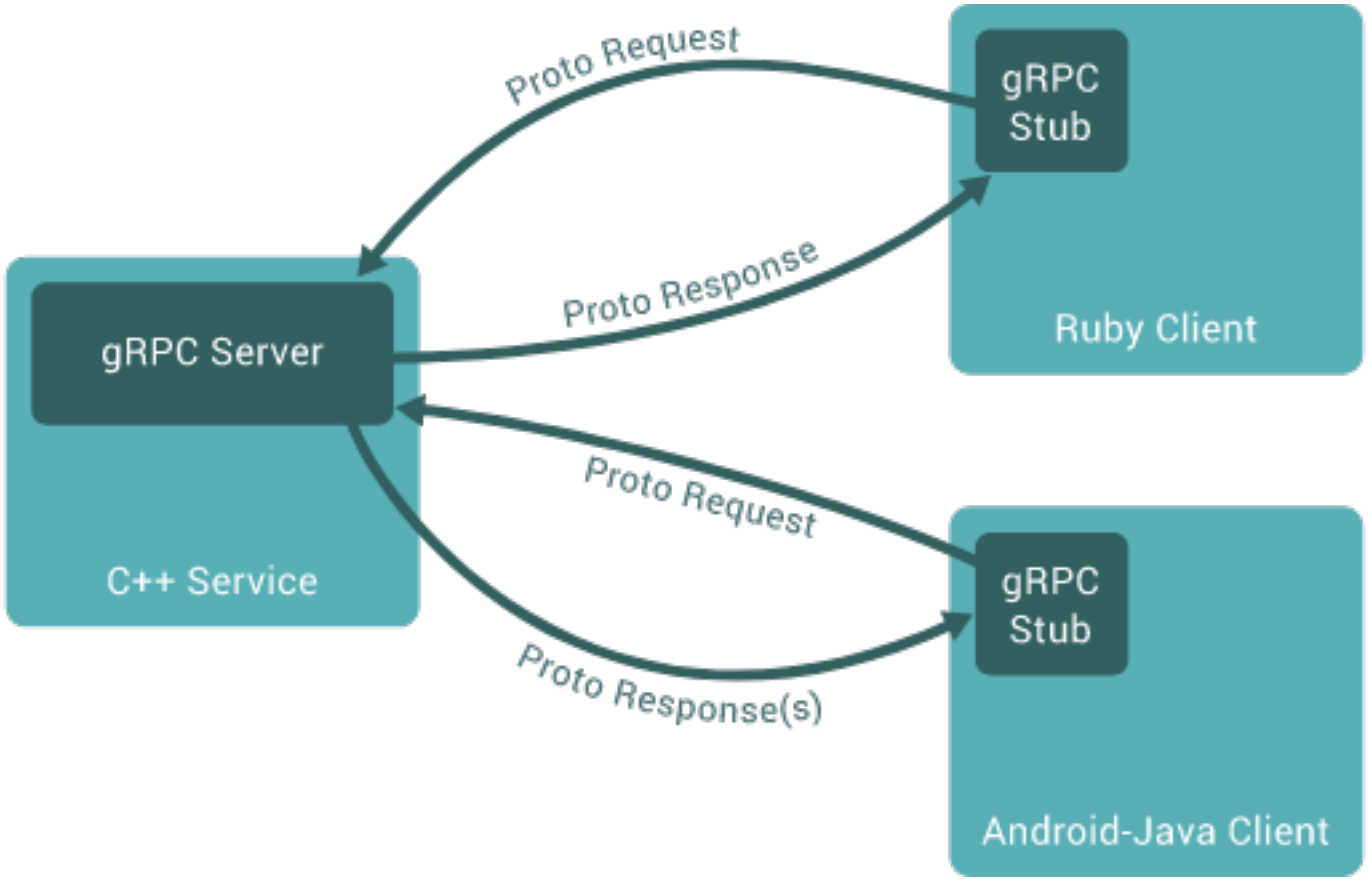
TensorFlow Serving
Client

```
hhp@HHPs:~/serving/tensorflow_serving/example$ python mnist_client.py --num_test
s=1000 --server=202.45.128.157:9000
Extracting /tmp/train-images-idx3-ubyte.gz
Extracting /tmp/train-labels-idx1-ubyte.gz
Extracting /tmp/t10k-images-idx3-ubyte.gz
Extracting /tmp/t10k-labels-idx1-ubyte.gz
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................
Inference error rate: 10.4%
```

TensorFlow Serving
Version transition

```
net@net-r4:~/0/serving$ tensorflow_model_server --port=9000 --model_name=mnist --model_base_path=/tmp/mnist_model/
2017-12-26 16:46:40.345223: I tensorflow_serving/model_servers/main.cc:147] Building single TensorFlow model file config:  model_name: mnist model_base_path: /tmp/mnist_model/
2017-12-26 16:46:40.345429: I tensorflow_serving/model_servers/server_core.cc:441] Adding/updating models.
2017-12-26 16:46:40.345464: I tensorflow_serving/model_servers/server_core.cc:492]  (Re-)adding model: mnist
2017-12-26 16:46:40.445851: I tensorflow_serving/core/basic_manager.cc:705] Successfully reserved resources to load servable {name: mnist version: 1}
2017-12-26 16:46:40.445887: I tensorflow_serving/core/loader_harness.cc:66] Approving load for servable version {name: mnist version: 1}
2017-12-26 16:46:40.445904: I tensorflow_serving/core/loader_harness.cc:74] Loading servable version {name: mnist version: 1}
2017-12-26 16:46:40.445934: I external/org_tensorflow/tensorflow/contrib/session_bundle/bundle_shim.cc:360] Attempting to load native SavedModelBundle in bundle-shim from: /tmp/m
nist_model/1
2017-12-26 16:46:40.445966: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:236] Loading SavedModel from: /tmp/mnist_model/1
2017-12-26 16:46:40.447744: I external/org_tensorflow/tensorflow/core/platform/cpu_feature_guard.cc:137] Your CPU supports instructions that this TensorFlow binary was not compil
ed to use: AVX2 FMA
2017-12-26 16:46:40.482343: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:155] Restoring SavedModel bundle.
2017-12-26 16:46:40.486518: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:190] Running LegacyInitOp on SavedModel bundle.
2017-12-26 16:46:40.489666: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:284] Loading SavedModel: success. Took 43689 microseconds.
2017-12-26 16:46:40.489942: I tensorflow_serving/core/loader_harness.cc:86] Successfully loaded servable version {name: mnist version: 1}
E1226 16:46:40.492883438    4998 ev_epoll1_linux.c:1051]     grpc epoll fd: 3
2017-12-26 16:46:40.494867: I tensorflow_serving/model_servers/main.cc:288] Running ModelServer at 0.0.0.0:9000 ...
2017-12-26 16:54:24.410098: I tensorflow_serving/core/basic_manager.cc:705] Successfully reserved resources to load servable {name: mnist version: 2}
2017-12-26 16:54:24.410149: I tensorflow_serving/core/loader_harness.cc:66] Approving load for servable version {name: mnist version: 2}
2017-12-26 16:54:24.410162: I tensorflow_serving/core/loader_harness.cc:74] Loading servable version {name: mnist version: 2}
2017-12-26 16:54:24.410188: I external/org_tensorflow/tensorflow/contrib/session_bundle/bundle_shim.cc:360] Attempting to load native SavedModelBundle in bundle-shim from: /tmp/m
nist_model/2
2017-12-26 16:54:24.410206: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:236] Loading SavedModel from: /tmp/mnist_model/2
2017-12-26 16:54:24.414058: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:155] Restoring SavedModel bundle.
2017-12-26 16:54:24.418692: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:190] Running LegacyInitOp on SavedModel bundle.
2017-12-26 16:54:24.422719: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:284] Loading SavedModel: success. Took 12497 microseconds.
2017-12-26 16:54:24.423051: I tensorflow_serving/core/loader_harness.cc:86] Successfully loaded servable version {name: mnist version: 2}
2017-12-26 16:54:24.510008: I tensorflow_serving/core/loader_harness.cc:137] Quiescing servable version {name: mnist version: 1}
2017-12-26 16:54:24.510078: I tensorflow_serving/core/loader_harness.cc:144] Done quiescing servable version {name: mnist version: 1}
2017-12-26 16:54:24.510090: I tensorflow_serving/core/loader_harness.cc:119] Unloading servable version {name: mnist version: 1}
2017-12-26 16:54:24.511401: I ./tensorflow_serving/core/simple_loader.h:294] Calling MallocExtension_ReleaseToSystem() after servable unload with 60742
2017-12-26 16:54:24.511430: I tensorflow_serving/core/loader_harness.cc:127] Done unloading servable version {name: mnist version: 1}
```

TensorFlow Serving
gRPC



TensorFlow Serving

Example: TensorFlow Serving

```
net@net-r4:~/0$ ls /tmp/mnist_model/
1
net@net-r4:~/0$ tensorflow_model_server --port=9000 --model_name=mnist --model_base_path=/tmp/mnist_model
2017-12-20 14:45:12.847054: I tensorflow_serving/model_servers/main.cc:147] Building single TensorFlow model file config:  model_name: mnist model_base_path: /tmp/mnist_model
2017-12-20 14:45:12.847264: I tensorflow_serving/model_servers/server_core.cc:441] Adding/updating models.
2017-12-20 14:45:12.847292: I tensorflow_serving/model_servers/server_core.cc:492]  (Re-)adding model: mnist
2017-12-20 14:45:12.947740: I tensorflow_serving/core/basic_manager.cc:705] Successfully reserved resources to load servable {name: mnist version: 1}
2017-12-20 14:45:12.947778: I tensorflow_serving/core/loader_harness.cc:66] Approving load for servable version {name: mnist version: 1}
2017-12-20 14:45:12.947806: I tensorflow_serving/core/loader_harness.cc:74] Loading servable version {name: mnist version: 1}
2017-12-20 14:45:12.947839: I external/org_tensorflow/tensorflow/contrib/session_bundle/bundle_shim.cc:360] Attempting to load native SavedModelBundle in bundle-shim from: /tmp/mnist_model/1
2017-12-20 14:45:12.947862: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:236] Loading SavedModel from: /tmp/mnist_model/1
2017-12-20 14:45:12.949489: I external/org_tensorflow/tensorflow/core/platform/cpu_feature_guard.cc:137] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2 FMA
2017-12-20 14:45:12.981194: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:155] Restoring SavedModel bundle.
2017-12-20 14:45:12.985189: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:190] Running LegacyInitOp on SavedModel bundle.
2017-12-20 14:45:12.988129: I external/org_tensorflow/tensorflow/cc/saved_model/loader.cc:284] Loading SavedModel: success. Took 40256 microseconds.
2017-12-20 14:45:12.988360: I tensorflow_serving/core/loader_harness.cc:86] Successfully loaded servable version {name: mnist version: 1}
E1220 14:45:12.991214483   14098 ev_epoll1_linux.c:1051]     grpc epoll fd: 3
2017-12-20 14:45:12.993514: I tensorflow_serving/model_servers/main.cc:288] Running ModelServer at 0.0.0.0:9000 ...
```

```
net@net-r4:~/0/serving$ python tensorflow_serving/example/mnist_saved_model.py --model_version=2 /tmp/mnist_model/
Training model...
Extracting /tmp/train-images-idx3-ubyte.gz
Extracting /tmp/train-labels-idx1-ubyte.gz
Extracting /tmp/t10k-images-idx3-ubyte.gz
Extracting /tmp/t10k-labels-idx1-ubyte.gz
2017-12-20 14:49:31.500257: I tensorflow/core/platform/cpu_feature_guard.cc:137] Your CPU supports instructions that this TensorFlow binary was not compiled to use: SSE4.1 SSE4.2 AVX AVX2 FMA
2017-12-20 14:49:31.680649: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1030] Found device 0 with properties:
name: GeForce GTX 1080 major: 6 minor: 1 memoryClockRate(GHz): 1.835
```

TensorFlow Lite

TensorFlow's **lightweight** solution for mobile and embedded devices.

- **Lightweight** Enables inference of on-device machine learning models with a small binary size and fast initialization/startup
- **Cross-platform** A runtime designed to run on many different platforms, starting with Android and iOS
- **Fast** Optimized for mobile devices, including dramatically improved model loading times, and supporting hardware acceleration

https://developers.googleblog.com/2017/11/announcing-tensorflow-lite.html

TensorFlow Lite

Components：

- A set of core operators which have been tuned for mobile platforms.
- FlatBuffers, an open-sourced, efficient cross platform serialization library.
- A new mobile-optimized interpreter.
- Android Neural Networks library for leveraging hardware acceleration.

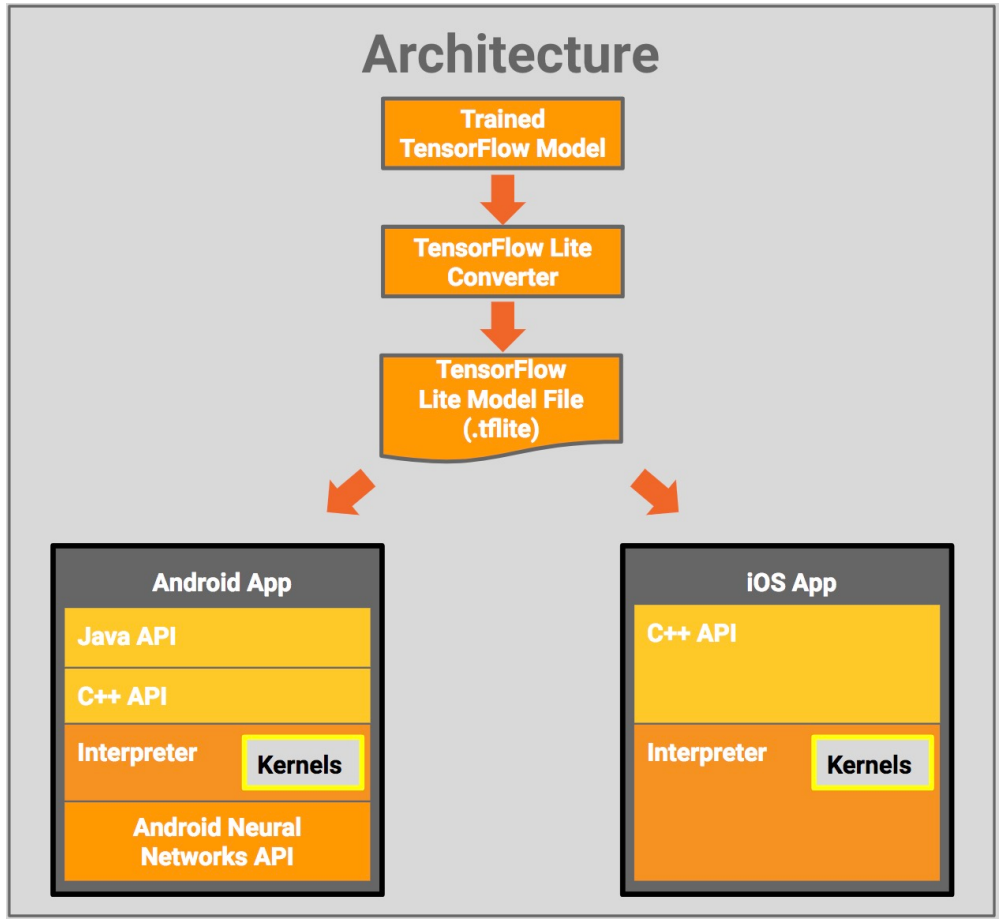
Motivation

- Interest in stronger user data privacy paradigms where user data does not need to leave the mobile device.
- Ability to serve ‘offline’ use cases, where the device does not need to be connected to a network.

https://www.tensorflow.org/mobile/tflite/

TensorFlow Lite
Architecture

- **TensorFlow Model**: A trained TensorFlow model saved on disk.
- **TensorFlow Lite Converter**: A program that converts the model to the TensorFlow Lite file format.
- **TensorFlow Lite Model File**: A model file format based on FlatBuffers, that has been optimized for maximum speed and minimum size.



https://developers.googleblog.com/2017/11/announcing-tensorflow-lite.html

TensorFlow Lite

TensorFlow Lite versus TensorFlow Mobile

- TensorFlow Lite is an **evolution** of TensorFlow Mobile. In most cases, apps developed with TensorFlow Lite will have a smaller binary size, fewer dependencies, and better performance.
- TensorFlow Lite is in developer preview, so not all use cases are covered yet. We expect you to use TensorFlow Mobile to cover production cases.
- TensorFlow Lite supports only a limited set of operators, so not all models will work on it by default. TensorFlow for Mobile has a fuller set of supported functionality.

https://www.tensorflow.org/mobile/

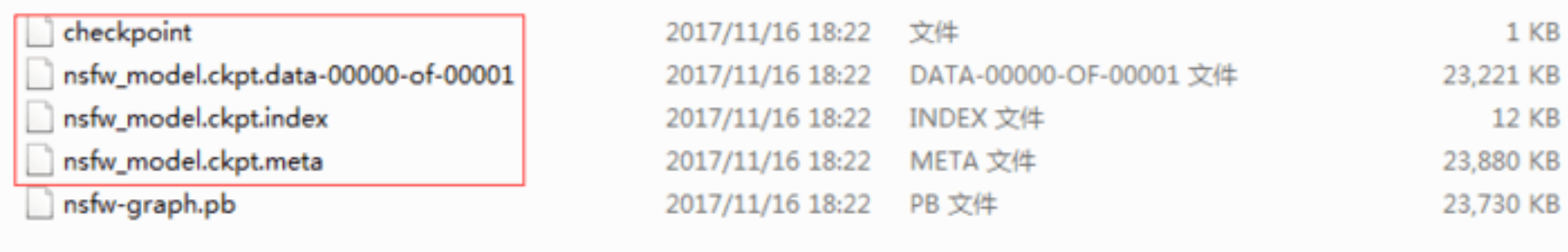
TensorFlow Lite

Convert a trained model to a .tflite file

1 Export GraphDef files(usually ending with the .pb or .pbtxt extension) and checkpoint files

```
saver = tf.train.Saver()

with tf.Session() as sess:
    sess.run(tf.global_variables_initializer())
    #使用tf.train.write_graph导出GraphDef文件
    tf.train.write_graph(sess.graph_def, "model/", 'nsfw-graph.pb', as_text=False)
    #使用tf.train.saver导出checkpoint文件
    saver.save(sess, "model/nsfw_model.ckpt")
```

- **write_graph => GraphDef =>** define the computational graph to execute.
- **save => checkpoint files** => record current values of the model variables.



https://www.jianshu.com/p/fa204a54a956?from=singlemessage

TensorFlow Lite

Convert a trained model to a .tflite file

2 Generate a “frozen” file

Freeze_graph

One of the Graph Transform Tools of TensorFlow Lite

```
bazel-bin/tensorflow/python/tools/freeze_graph \
--input_graph=/data/deep_learning/nsfw/model/nsfw-graph.pb \
--input_checkpoint=/data/deep_learning/nsfw/model/nsfw_model.ckpt \
--input_binary=true \
--output_graph=/data/deep_learning/nsfw/model/frozen_nsfw.pb \
--output_node_names=predictions
```

https://www.jianshu.com/p/fa204a54a956?from=singlemessage

TensorFlow Lite

Convert a trained model to a .tflite file

3 Generate a .tflite file

Toco

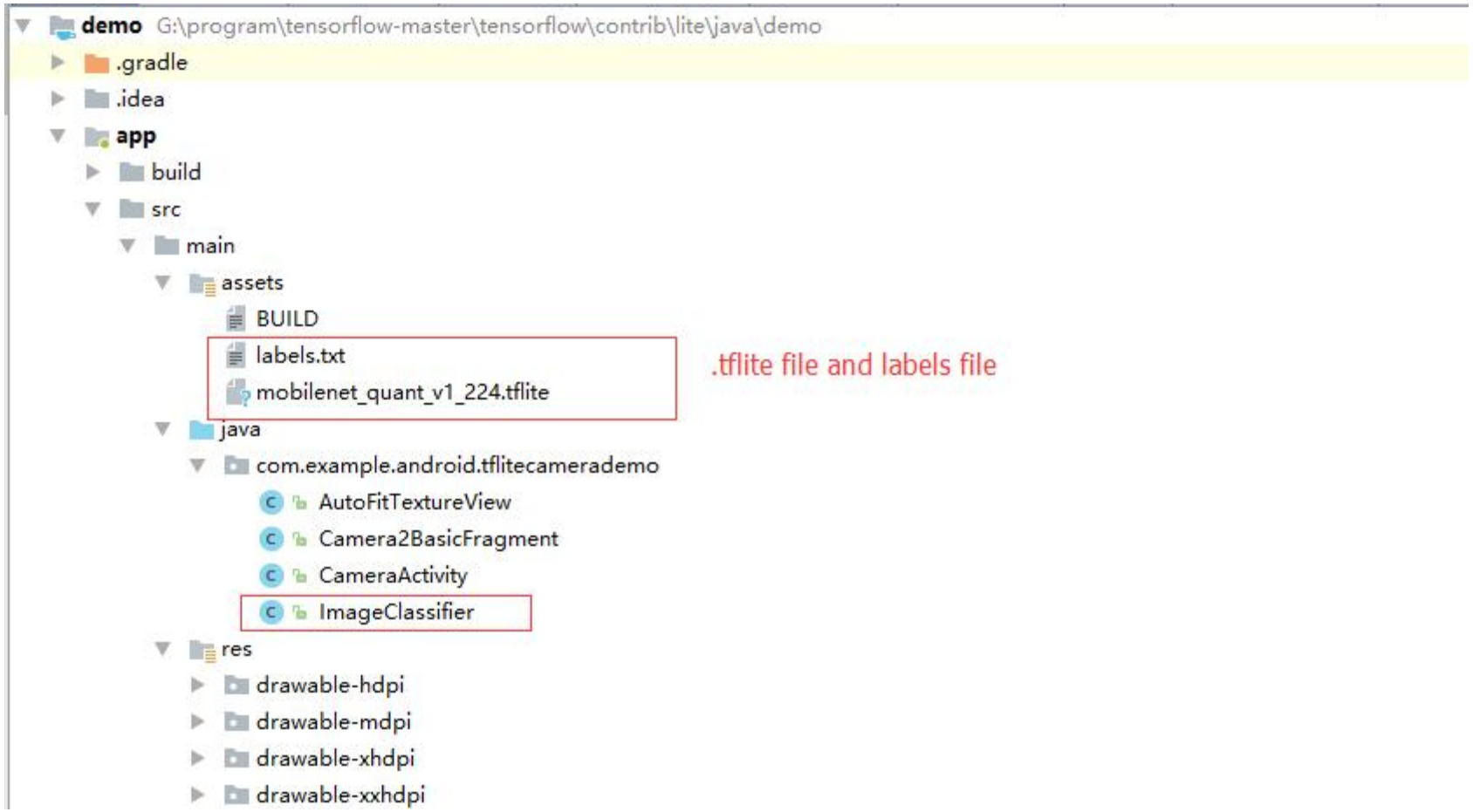
One of the Graph Transform Tools of TensorFlow Lite

```
bazel run --config=opt tensorflow/contrib/lite/toco:toco \
--input_file=/data/deep_learning/nsfw/model/frozen_nsfw.pb \
--input_format=TENSORFLOW_GRAPHDEF \
--output_format=TFLITE \
--output_file=/data/deep_learning/nsfw/model/nsfw.lite \
--inference_type=FLOAT \
--input_type=FLOAT \
--input_arrays=input \
--output_arrays=predictions \
--input_shapes=1,224,224,3
```

https://www.jianshu.com/p/fa204a54a956?from=singlemessage

TensorFlow Lite

Building TensorFlow on Android



TensorFlow Lite

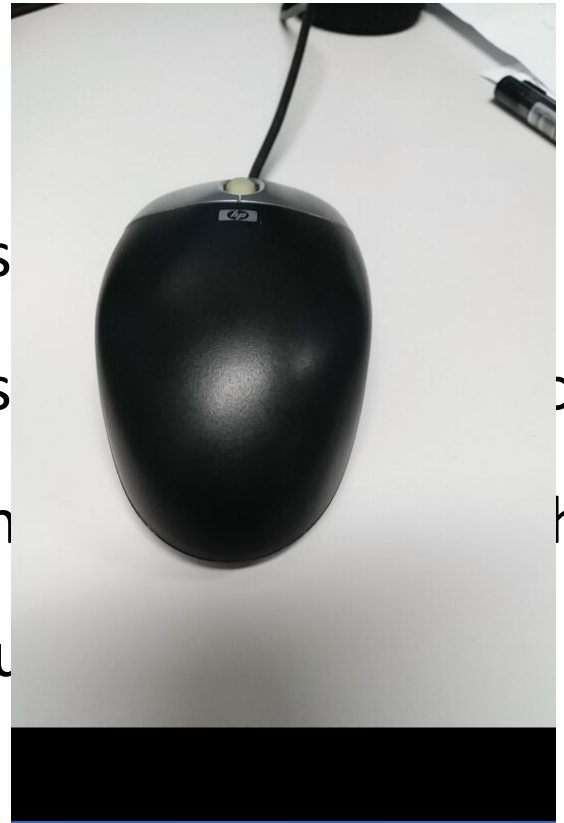
Building TensorFlow on Android

```
import org.tensorflow.lite.Interpreter;

/** Classifies images with Tensorflow Lite. */
public class ImageClassifier {

  /** Tag for the {@link Log}. */
  private static final String TAG = "TfLiteCameraDemo";

  /** Name of the model file stored in Assets. */
  private static final String MODEL_PATH = "mobilenet_quant_v1_224.tflite";

  /** Name of the label file stored in Assets. */
  private static final String LABEL_PATH = "labels.txt";

  /** Number of results to show in the UI. */
  private static final int RESULTS_TO_SHOW = 3;
```

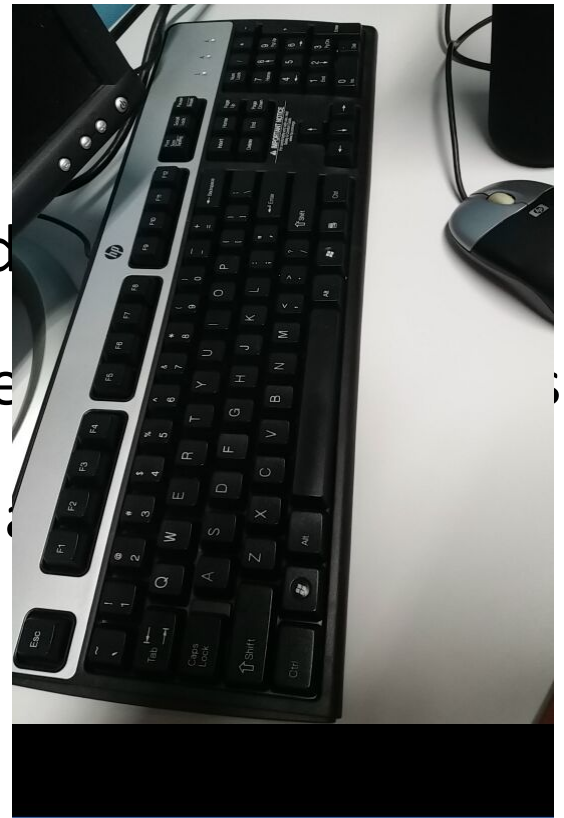
TensorFlow Lite

Example: object recognition

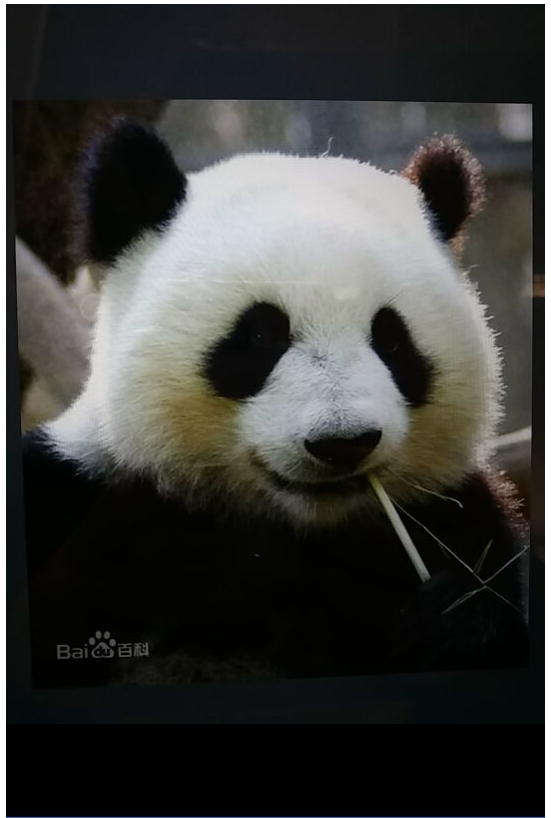
106ms
mouse:1.0
goldfish:0.0
toilet tissue:0.0



92ms
mouse:0.90588236
punching bag:0.0627451
wok:0.003921569



93ms
computer keyboard:0.6862745
laptop:0.15294118
space bar:0.05882353



122ms
giant panda:0.99607843
lesser panda:0.007843138
toilet tissue:0.0

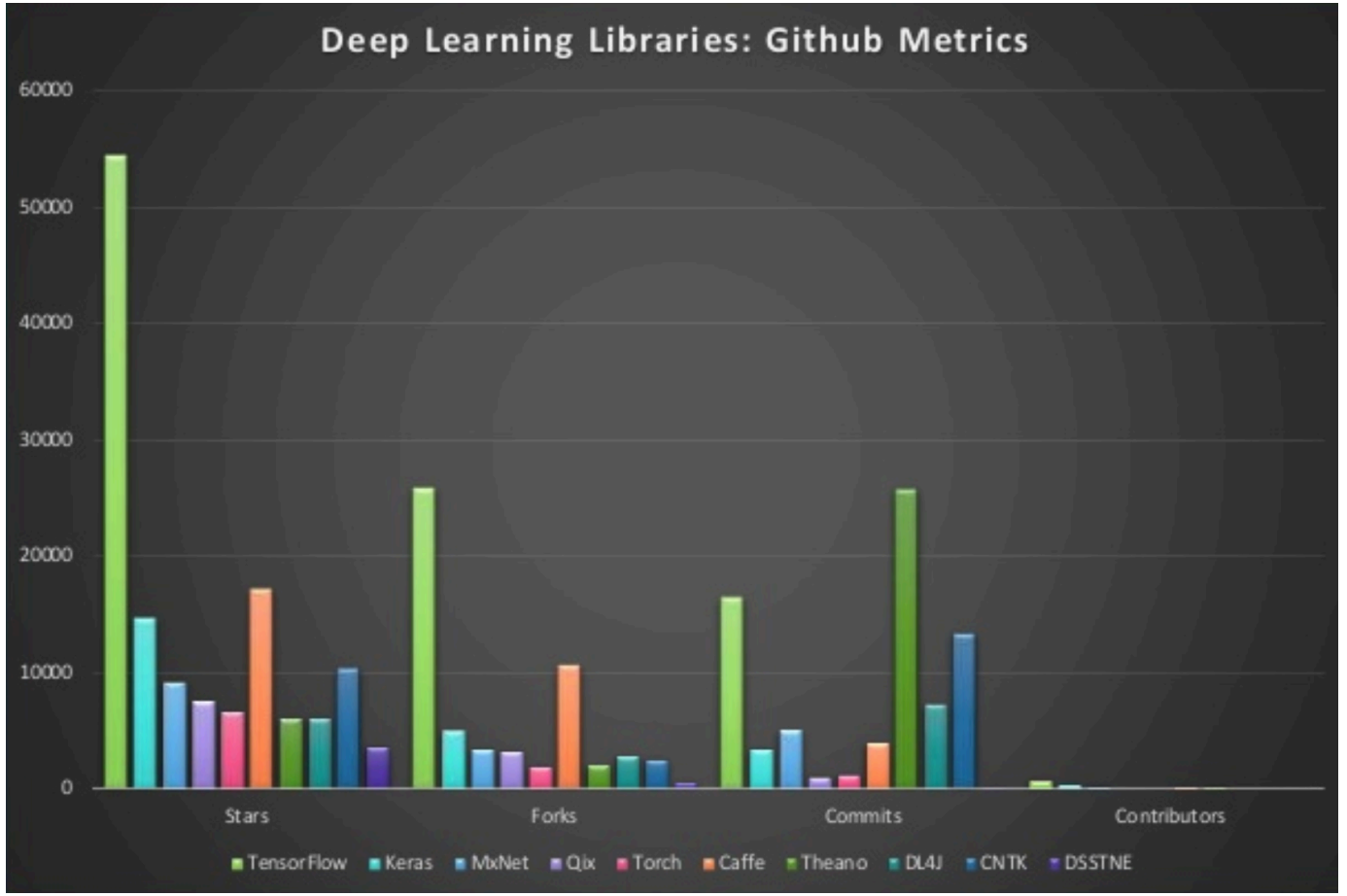
Comparison

Table 1: Open-source Frameworks

Platform	Tensorflow	CNTK	Deeplearning4j	MXNet	H2O	Caffe	Theano	Torch
Release Date	2016	2016	2015	2015	2014	2014	2010	2011 (deep learning)
Core Language	C++	C++	C++	C++	Java	C++	C++	C
API	C++, Python	NDL	Java, Scala	C++, Python, R, Scala, Matlab, Javascript, Go, Julia	Java, R, Python, Scala, Javascript, web-UI	Python, Matlab	Python	Lua
Synchronization Model	Sync or async	Sync	Sync	Sync or async	Async	Sync	Async	Sync
Communication Model	Parameter server	MPI	Iterative MapReduce	Parameter server	Distributed fork-join	N/A	N/A	N/A
Multi-GPU	✓	✓	✓	✓	✓	✓	✓	✓
Multi-node	✓	✓	✓	✓	✓	✗	✗	✗
Data Parallelism	✓	✓	✓	✓	✓	✓	✓	✓
Model Parallelism	✓	N/A	✗	✓	✗	✗	✓	✓
Deep Learning Models	DBN, CNN, RNN	DBN, CNN, RNN	DBN, CNN, RNN	DBN, CNN, RNN	DBN	DBN, CNN, RNN	DBN, CNN, RNN	DBN, CNN, RNN
Programming Paradigm	Imperative	Imperative	Declarative	Both	Declarative	Declarative	Imperative	Imperative
Fault Tolerance	Checkpoint-and-recovery	Checkpoint-and-resume	Checkpoint-and-resume	Checkpoint-and-resume	N/A	N/A	Checkpoint-and-resume	Checkpoint-and-resume
Visualization	Graph (interactive), training monitoring	Graph (static)	Training monitoring	None	None	Summary Statistics	Graph (static)	Plots

http://dsc.soic.indiana.edu/publications/DLFrameworks.pdf

GITHUB VIEW



https://www.slideshare.net/AshishBansal17/tensorflow-vs-mxnet

Conclusion

TensorFlow

Focus on training

TensorFlow Serving

Focus on learning and serving trained models

TensorFlow Lite

Focus on learning on mobile-devices

THANK YOU