Outline

- Motivation
- What is TensorFlow
- Under the Hood
- Machine Learning in TensorFlow
- Updates
- Ecosystem
- Resources
Motivation
### Results with Deep Learning

<table>
<thead>
<tr>
<th><strong>Input</strong></th>
<th><strong>Output</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Pixels:</td>
<td>&quot;lion&quot;</td>
</tr>
<tr>
<td>Audio:</td>
<td>&quot;see at tuh</td>
</tr>
<tr>
<td>&lt;query, doc&gt;</td>
<td>P(click on doc)</td>
</tr>
<tr>
<td>&quot;Hello, how are you?&quot;</td>
<td>&quot;Bonjour, comment allez-vous?&quot;</td>
</tr>
<tr>
<td>Pixels:</td>
<td>&quot;A close up of a small child holding a stuffed animal&quot;</td>
</tr>
</tbody>
</table>
Growing Use of Deep Learning at Google

Across many products/areas
- Apps
- Maps
- Photos
- Gmail
- Speech
- Android
- YouTube
- Translation
- Robotics Research
- Image Understanding
- Natural Language Understanding
- Drug Discovery
Products using Machine Learning
What is TensorFlow?
TensorFlow: What is it?

- Dataflow graph-based computational infrastructure
  - General purpose.
    - Great for DeepLearning
  - Distributed
  - Parallel
- Heterogeneous devices
  - Data Centers
  - Desktops with CPUs and GPUs
  - Mobile devices
TensorFlow: Who is it for?

- Researchers: Publish papers with real code for faster collaboration
- Data Scientists: Rapidly growing library of best in class models
- Developers: Same API used for training and deployment
- Everybody with interests in machine learning
TensorFlow: Architecture

High-level Libraries
- Model Library
- TensorBoard
- Other Languages

TensorFlow Core
- Library of General Use Ops
- Neural Net Ops
- Distributed, Parallel Graph Execution Engine

Platforms
- CPU
- GPU
- TPU
- Android
- iOS

Ops
- Neural Net

Languages
- Python Frontend
- C++ Frontend
- Other Languages
Under the Hood
Computation as a Dataflow Graph

Graph of *Nodes*, also called *Operations* (ops)

- biases
- weights
- inputs
- targets
- MatMul
- Add
- Softmax
- Xent
Dataflow Graph (forward)

Edges are N-dimensional arrays: **Tensors**

- biases
- weights
- inputs
- targets
- MatMul
- Add
- Softmax
- Xent
Dataflow Graph (backward graph and updates)

Backward graph and update are added automatically to graph

'Biases' is a variable

Some ops compute gradients

== updates biases

biases

... Add ...

... Mul

learning rate

==
Distributed Dataflow Graph

- Distribution across: Threads, Processes, GPUs, Machines, ...
- Many operations can be executed in parallel

![Distributed Dataflow Graph Diagram](image-url)
Parallelism

MatMul

Input

Param

MatMul

MatMul

...
● In practice often very complex with 100s or 1000s of nodes and edges
● “Inference” means to execute just the forward path of the graph
Machine Learning in TensorFlow
Example: Logistic Regression

\[
\sum_{i=0}^{m} \log(1 + e^{-t(wx_0 + b)})
\]
General Purpose Operations

- **Basics:** constant, random, placeholder, cast, shape
- **Variables:** assign, assign_sub, assign_add
- **Queues:** enqueue, enqueue_batch, dequeue, blocking or not.
- **Logical:** equal, greater, less, where, min, max, argmin, argmax.
- **Tensor computations:** all math ops, matmul, determinant, inverse, cholesky.
- **Images:** encode, decode, crop, pad, resize, color spaces, random perturbations.
- **Sparse tensors:** represented as 3 tensors.
- **File IO:** file reader (sstable, recordio, txt, csv), parser for Example protocol buffers.
- **Control flow:** control dependencies, conditionals, loops, functions.
Neural Net Operations

- **Activations**: sigmoid, tanh, relu, relu6, elu, dropout.
- **Pooling**: avg, max.
- **Convolutions**: 1D, 2D, 3D with many options.
- **Normalization**: local, batch, moving averages.
- **Classification**: softmax, softmax loss, cross entropy loss, topk.
- **Embeddings**: distributed lookups/gather, scatter/updates.
- **Sampling**: candidate sampler (various options), sampling softmax.
- **Updates**: "fused ops" to speed-up optimizer updates (Adagrad, Momentum.)
- **Summaries**: Capture information for visualization.
Batch Logistic Regression in TensorFlow: Graph Construction

```python
graph = tf.Graph()
with graph.as_default():
    examples = tf.constant(train_dataset)
    labels = tf.constant(train_labels)

    W = tf.Variable(tf.truncated_normal(
        [image_size * image_size, num_labels]))
    b = tf.Variable(tf.zeros([num_labels]))
    logits = tf.matmul(examples, W) + b

    xent = tf.nn.softmax_cross_entropy_with_logits(
        logits, labels)
    loss = tf.reduce_mean(xent)

    optimizer = tf.train.GradientDescentOptimizer(0.5).minimize(loss)

    prediction = tf.nn.softmax(logits)
```

# Create new computation graph

# Training data/labels

# Hidden layer

# Loss

# Optimizer to minimize loss

# Predictions
Batch Logistic Regression in TensorFlow (Computation)

```python
graph = tf.Graph()
with graph.as_default():
    examples = tf.constant(train_dataset)
    labels = tf.constant(train_labels)
    W = tf.Variable(tf.truncated_normal([image_size * image_size, num_labels]))
    b = tf.Variable(tf.zeros([num_labels]))
    logits = tf.matmul(examples, W) + b
    xent = tf.nn.softmax_cross_entropy_with_logits(logits, labels)
    loss = tf.reduce_mean(xent)
    optimizer = tf.train.GradientDescentOptimizer(0.5).minimize(loss)
    prediction = tf.nn.softmax(logits)

with tf.Session(graph=graph) as session:
    session.run(tf.initialize_all_variables())
    for step in xrange(num_steps):
        _, l, predictions = session.run([optimizer, loss, prediction])
        if (step % 100 == 0):
            print 'Loss at step', step, ':', l
        print 'Training accuracy: %.1f%%' % accuracy(predictions, labels)
```

# Create a session

# Train for a number of steps
Example: K-Means clustering

\[
\arg\min_S \sum_{i=1}^{k} \sum_{x \in S_i} \|x - \mu_i\|^2
\]
K-Means in TensorFlow

# 'x' are the points to be clustered, 'k' is number of clusters
centroids = tf.Variable(tf.random_normal([k, x.shape[1]]))
distances = tf.pdist2(x, centroids)
nearest = tf.argmin(distances, 1)
update = tf.assign(centroids, tf.segment_mean(x, nearest))

with tf.Session() as session:
    session.run(tf.initialize_all_variables())
    for step in xrange(100):
        session.run(update)
Visualizing Learning

TensorBoard

input new regex
Split On Underscores:

X Type:
STEP  RELATIVE  WALL

Selected Runs:
☑ data

xentropy

xentropy_mean

0.000  400.0  800.0  1.200k  1.60k
Visualizing TensorFlow Graphs
Updates
Over the last 5 months

- Python 3.3+ support
- CUDA support extended to 7.5
- Moved to CuDNN R4
- Distributed (multi-machine) support based on gRPC
- Graph packaging for inference
- More high level Python APIs (slim, skflow, learn)
- iOS support
Ecosystem
Computation using data flow graphs for scalable machine learning [http://tensorflow.org](http://tensorflow.org)
“Neural Art” in TensorFlow

github.com/woodrush/neural-art-tf
Recurrent Net Dreams up Fake Chinese Characters
Deep-Q learning Pong with TensorFlow and PyGame
Deep Learning with Spark and TensorFlow

Tim Hunter, DataBricks
Models

- Inception - ImageNet model
- Autoencoders - Variational, Denoising
- Neural GPU - Learn Algorithms
- Swivel - Embeddings

github.com/tensorflow/models
Managed Scalable Machine Learning Platform

- Prediction at Scale
- Build Machine Learning Models Easily
- Fully Managed Platform
- Powered by TensorFlow
Google Cloud Machine Learning APIs

Image Analysis

Speech Recognition

Translation
Resources
Resources

Website: https://www.tensorflow.org

Code: https://github.com/tensorflow


Google Cloud ML: https://cloud.google.com/ml/