Kevin Lin

ML Systems, Fall 2022

August 31, 2022

Tensorflow

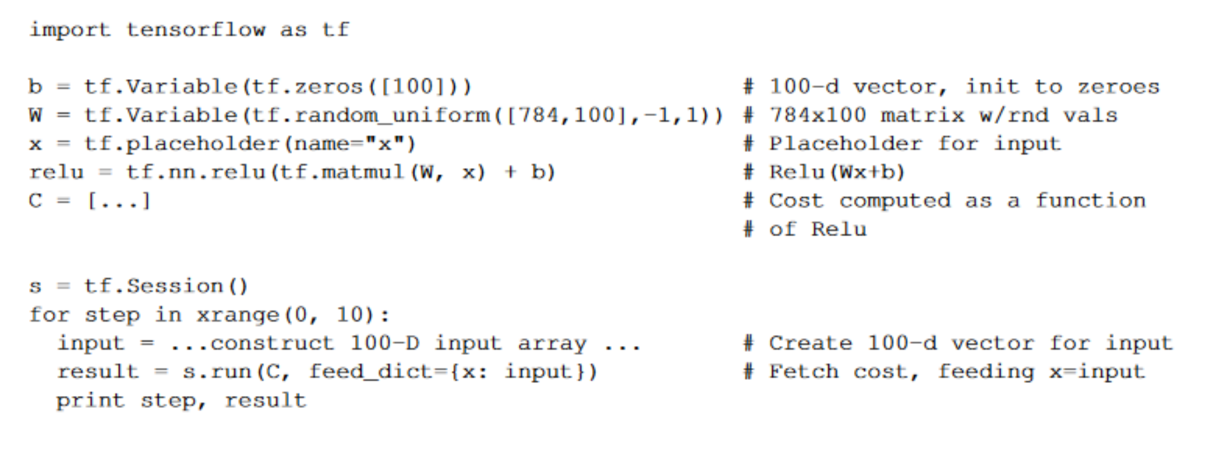
Overview

- **Open Source** Interface for expressing ML algorithms AND an implementation for executing these algorithms
- Built by Google in 2015 (initial release)
- Large amount of flexibility from mobile devices to HPCs
- Commonly used in Deep Learning to train models
  - Speech Recognition
  - Computer Vision
  - Robotics
  - Information Retrieval
  - Natural Language Processing (NLP)
- TensorFlow vs Pytorch?

Background

- Google Brain in 2011 explored very large deep neural networks
- DistBelief created as a scalable distributed training and inference system
- Research using DistBelief included
  - Unsupervised Learning
  - Language Representation
  - Image Classification and Object Detection
  - Video Classification
  - Speech Recognition
- Large adaptation in Google (50+ teams) and implemented in Google Search, Google Maps, YouTube, and Advertisements

Example



```
import tensorflow as tf

b = tf.Variable(tf.zeros([100]))                   # 100-d vector, init to zeroes
W = tf.Variable(tf.random_uniform([784,100],-1,1)) # 784x100 matrix w/rnd vals
x = tf.placeholder(name="x")                       # Placeholder for input
relu = tf.nn.relu(tf.matmul(W, x) + b)             # Relu(Wx+b)
C = [...]                                          # Cost computed as a function
                                                   # of Relu

s = tf.Session()
for step in xrange(0, 10):
  input = ...construct 100-D input array ...       # Create 100-d vector for input
  result = s.run(C, feed_dict={x: input})          # Fetch cost, feeding x=input
  print step, result
```

Figure 1: Example TensorFlow code fragment

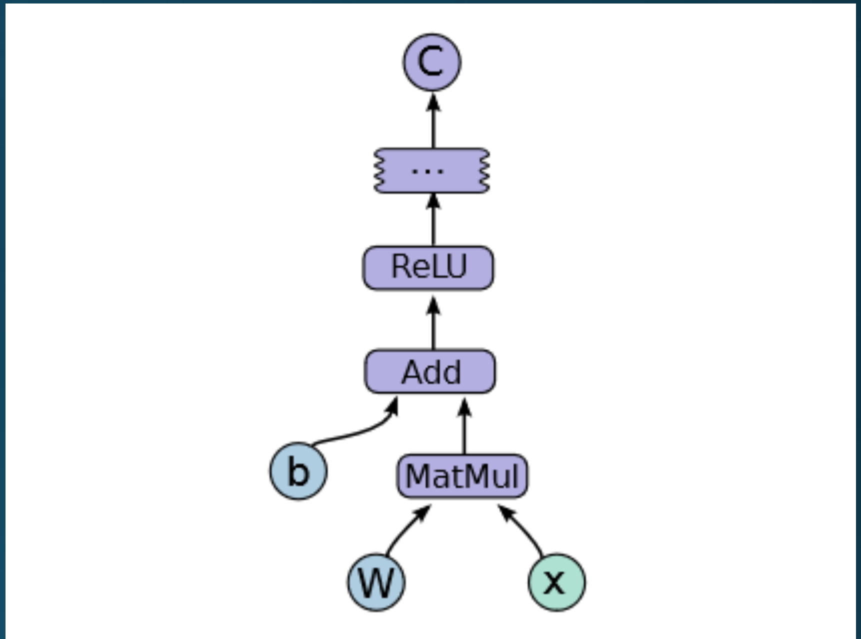


Figure 2: Corresponding computation graph for Figure 1

- TensorFlow Graph
- Nodes are instantiation of an operation
- Values are tensors

TensorFlow

- Single system that can span multiple platforms
  - Directly addresses previous issues with different systems used for training and deployment

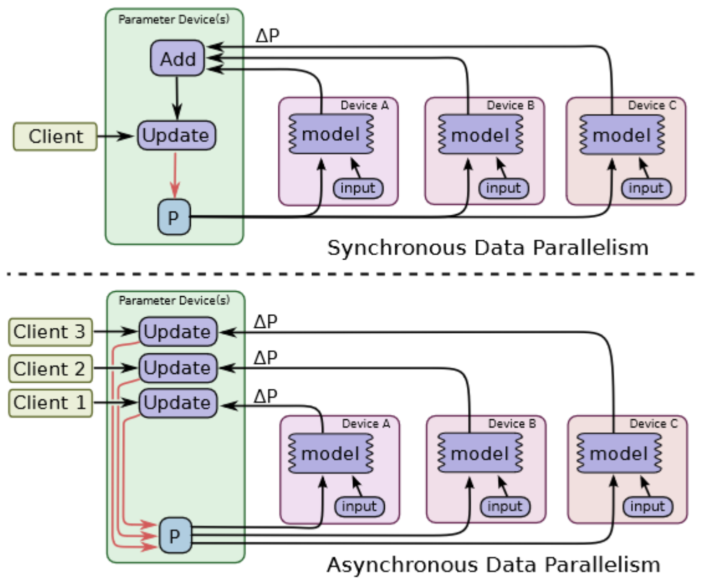
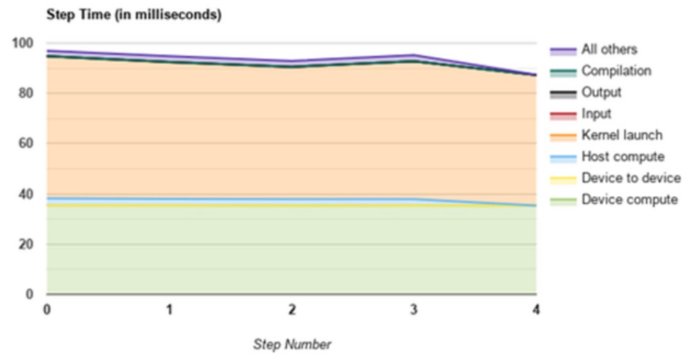


Figure 7: Synchronous and asynchronous data parallel training

Evaluation

- Single NVIDIA V100 , TensorFlow 2.2, Use tf.keras.model.fit() a nd tf.dataset APIs
- Optimize the time the training takes measured in number of samples being processed per second



Small Batch Size — tf profiler Step-time Graph (by author using TensorBoard)

- Half the time needed to load kernels to the GPU!

Advantages

- Open Source
- Adapts to all commonly used systems
- Proven scalability in numerous research areas and commonly used products
- Implements Keras which allows further capabilities particularly in deep learning
- Implements parallelism in work models which reduces memory allocation
- Tensor Processing Units (TPUs) perform computations faster than GPU or CPUs

Advantages

- Graphical Display of a CNN!

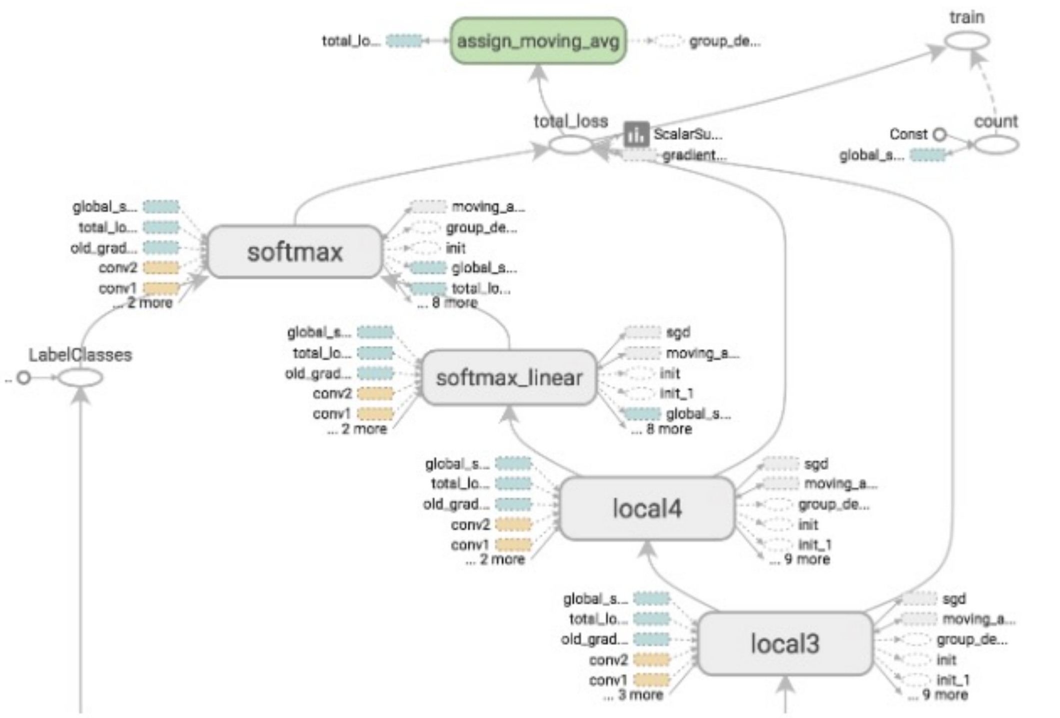


Figure 10: TensorBoard graph visualization of a convolutional neural network model

Disadvantages

- Frequent Updates (~2-3 months)
  - Verify version/environment before running someone else's code!
- Only supports NVIDIA for GPU and Python for GPU programming
- Comparatively slower compared to competitors (e.g. Pytorch)
- More features for Linux users versus Windows users
- Debugging can be difficult due to unique structure
- Steep learning curve (more backend code needed)
- As just a tool, can be used for anything!! (Ethical concerns)

Class Discussion