AutoDiff & TensorFlow

Deep RL Section 1/27/17

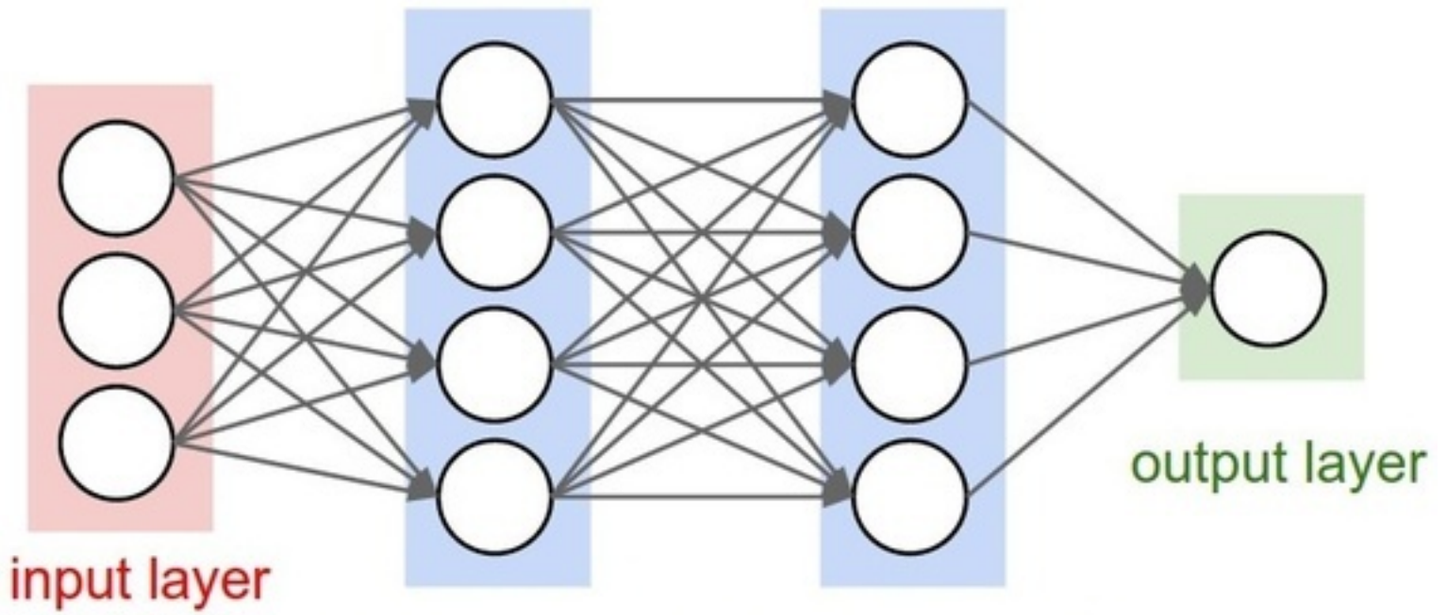
Chelsea Finn

Outline

1. Backpropagation and SGD
2. Automatic differentiation
3. Deep learning libraries, pros & cons
4. TensorFlow basics (**focus**)

— ask questions —

Backpropagation



$$\mathbf{h_1} = \sigma(W_1\mathbf{x}) \qquad y = W_3\mathbf{h_2}$$

$$\mathbf{h_2} = \sigma(W_2\mathbf{h_1})$$

$$\frac{\partial y}{\partial W_1} = \frac{\partial y}{\partial \mathbf{h_2}}\frac{\partial \mathbf{h_2}}{\partial \mathbf{h_1}}\frac{\partial \mathbf{h_1}}{\partial W_1} \qquad \frac{dy}{dW_3}$$

$$\frac{\partial y}{\partial W_2} = \frac{\partial y}{\partial \mathbf{h_2}}\frac{\partial \mathbf{h_2}}{\partial W_3}$$

picture from Stanford's CS231n

What is automatic differentiation?

1. You specify the computation graph by composing predefined functions
2. The program computes derivatives for you

Two models: forward mode autodiff & reverse-mode autodiff

Deep Learning Libraries

Most popular: TensorFlow, Caffe, Theano, Torch

Others: mxnet (Amazon & others), CNTK (Microsoft), chainer (PfNet), neon (Nervana)

TensorFlow (Google)
+ documentation, widely-used
very flexible, TensorBoard (viz)
- often somewhat slower

Caffe (UC Berkeley)
+ simple for standard nets, often fast
- lacking documentation, less flexible

Theano (U of Montreal)
+ widely-used, very flexible
- less stable, slow compile time

Torch (Facebook AI Research + others)
+ well-supported, among the fastest
- lua interface*

*until last week

TensorFlow: Installation

- supports python 2.7 and 3.3+
- CPU vs. GPU version, GPU requires CUDA and cuDNN
- v0.11 currently more stable than v0.12
- “pip install tensorflow” coming soon
- well-supported on Linux and Mac

TensorFlow: Main Idea

- **Variables, Tensors, Ops**
- Build symbolic computation graph in python
  - numpy-like syntax
  - **example operations**: log, matmul, conv2d, fft, sum, relu, sigmoid, dropout, concat, split, resize_bilinear, crop, random_saturation, batch_norm, softmax
- Execute graph or part of graph with session.run()
  - Usually call session.run() once for every training iteration
  - Only run parts of graph as necessary

TensorFlow: Input

- **Placeholders:** entry point of graph
  - enter data with feed_dict
- Built-in Data Readers: **TextLineReader**, **WholeFileReader**, **TFRecordReader** (protobuf)

Taking data in and out of TensorFlow tends to slow.

TensorFlow: Example

```
import tensorflow as tf

print 'Constructing Graph'
a = tf.placeholder("float")
b = tf.placeholder("float")
c = tf.get_variable('my_var', shape=(1), initializer=tf.constant_initializer(5))

y = tf.mul(a, tf.log(b)) + c

print 'Creating Session'
sess = tf.Session()

print 'Initializing Variables'
sess.run(tf.initialize_all_variables())

print 'Running Graph'
print sess.run(y, feed_dict={a: 3, b: 3})
```

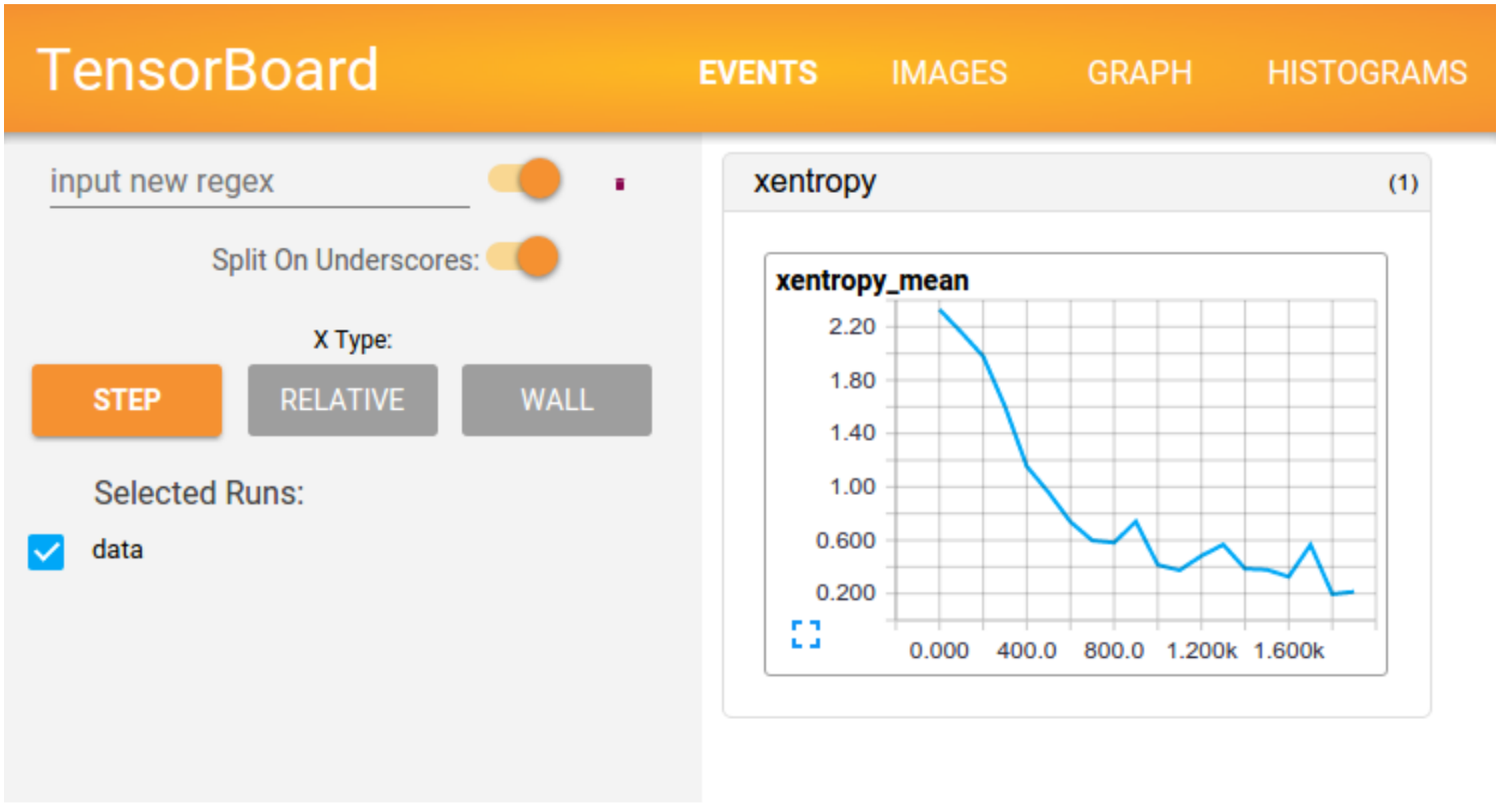
TensorFlow: Training

- **tf.train.Optimizer** (e.g. Momentum, Adagrad, Adam)
- feed in a tensor to be minimized

TensorFlow: Logging

- **Saver** object
  - for saving and restoring weights
  - define how many checkpoints to keep
- **SummaryWriter** - save summary of performance
  - tf.scalar_summary(), tf.image_summary()
- **TensorBoard** - automatically loads summaries and displays stats in browser, can easily run over ssh

TensorBoard



TensorFlow: Misc

- scoping
- reusing variables
- Advanced numpy-like array slicing not always supported
- Debugging:
  - pull values of tensors out of the graph
  - look at shapes (tensor.get_shape())
  - use InteractiveSession() to experiment in a shell

TensorFlow: Docs

- The docs are your friend!

  tensorflow.org/api_docs/python/

  [Useful for looking up graph ops]

- Source code also a useful reference

  github.com/tensorflow

TensorFlow: Add-ons / Wrappers

- **keras** - frontend wrapper, supports both Theano and TF backend, to become official TF fronted
- **tf-slim** - lightweight wrapper, reduce amount of code, works seamlessly with TF

Questions?