More Convolutional Neural Networks and an Introduction to Tensorflow

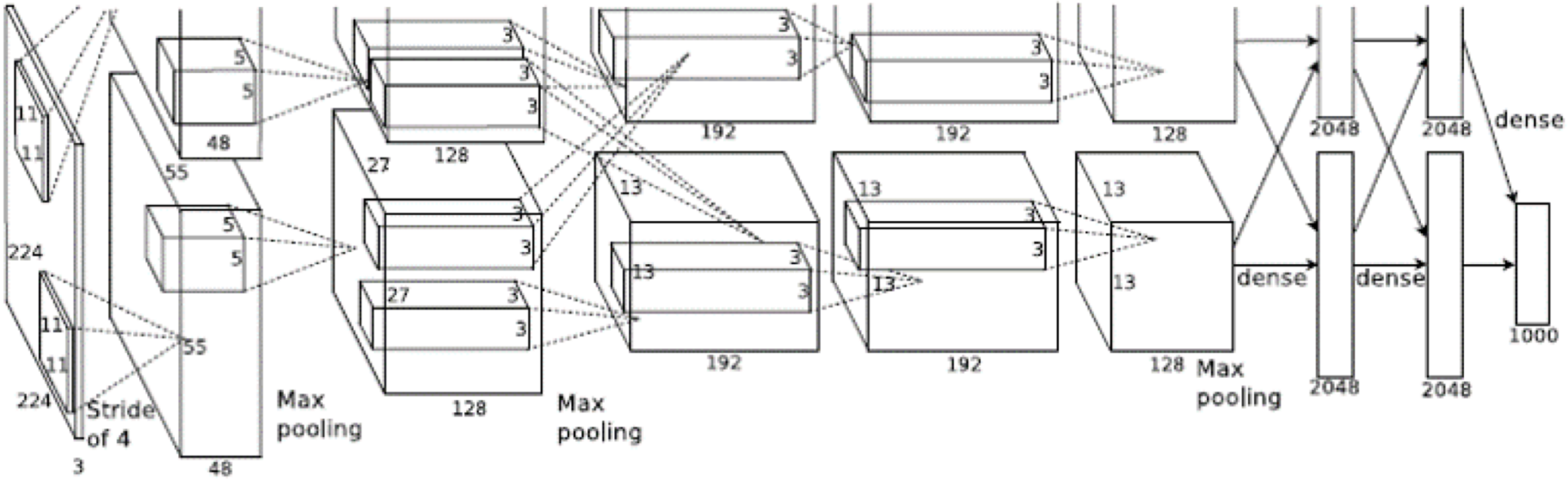
CS 510

Lecture #17

April 3rd, 2019

Colorado State University

AlexNet : The Start of a Revolution



Convolutional Neural Nets

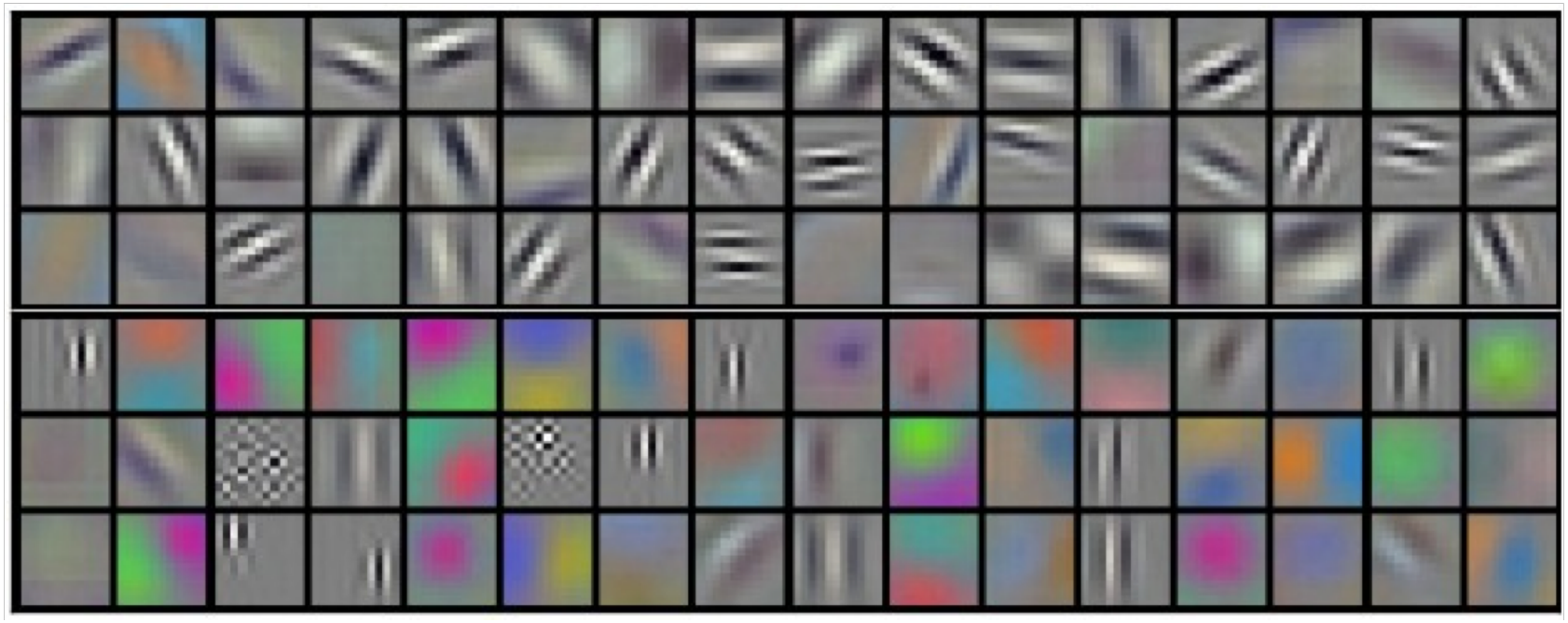
- Convolutional layers
  - Local, translation insensitive layers
  - Small number of re-used weights
- Pooling layers
  - Similar to image pyramid
  - No weights at all
- ReLu transfer function
  - Non-linear
  - Avoids vanishing derivatives

AlexNet Performance (2012)

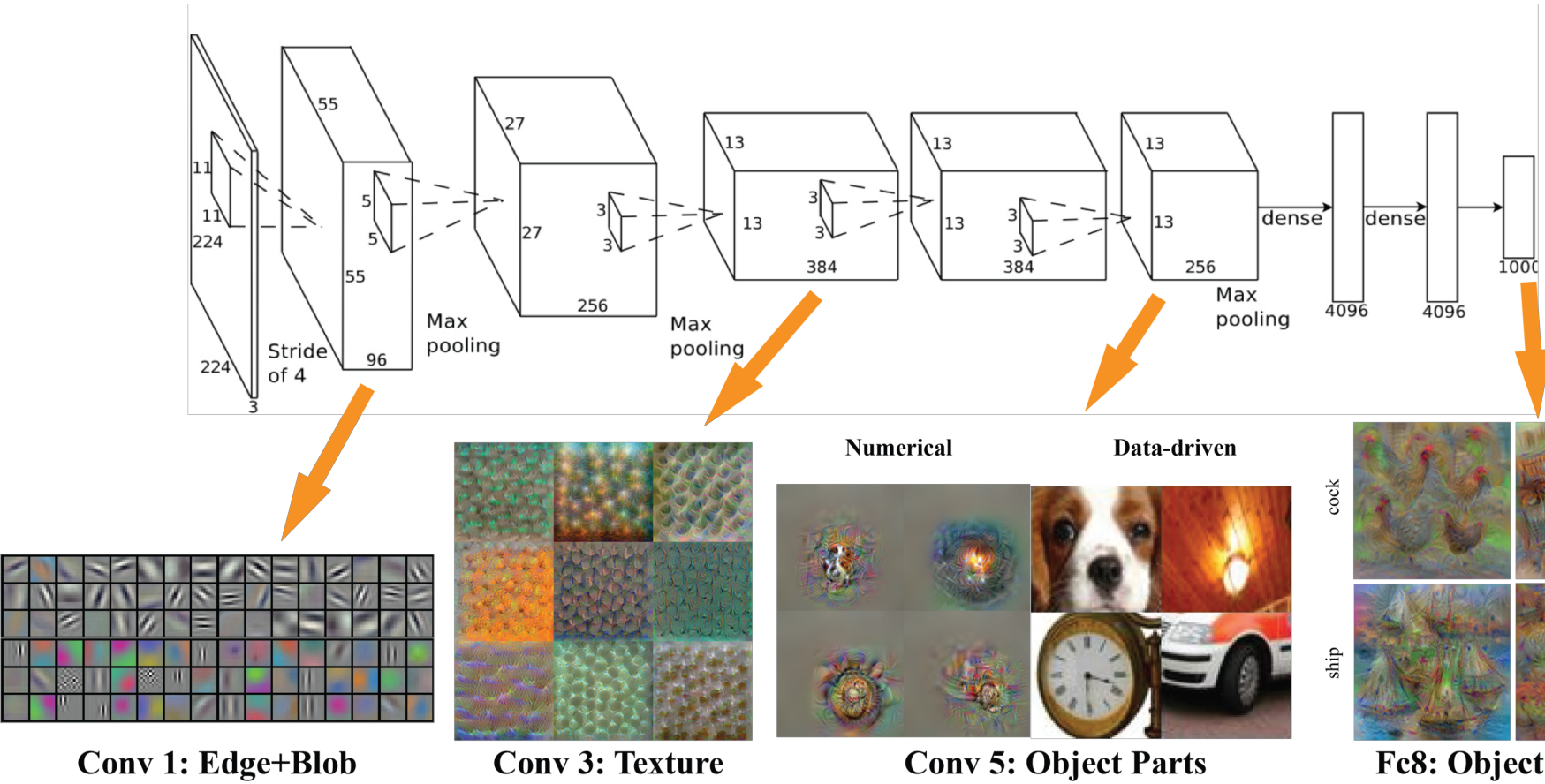
- Results reported for ILSCRV 2010
  - A test set based on ImageNet
  - 1000 image classes (random = 99.9% error rate)
- Results reported for Top-1 & Top-5
  - Output activation orders responses
  - Top-1 : best response only
  - Top-5 : is correct answer among top 5 responses?
- Error rates
  - Top-1 : 37.5% (62.5% correct)
  - Top-5 : 17% (83% answers within top 5)

What Does AlexNet Learn?

- Layer #1 Convolution masks:



What about other AlexNet layers?



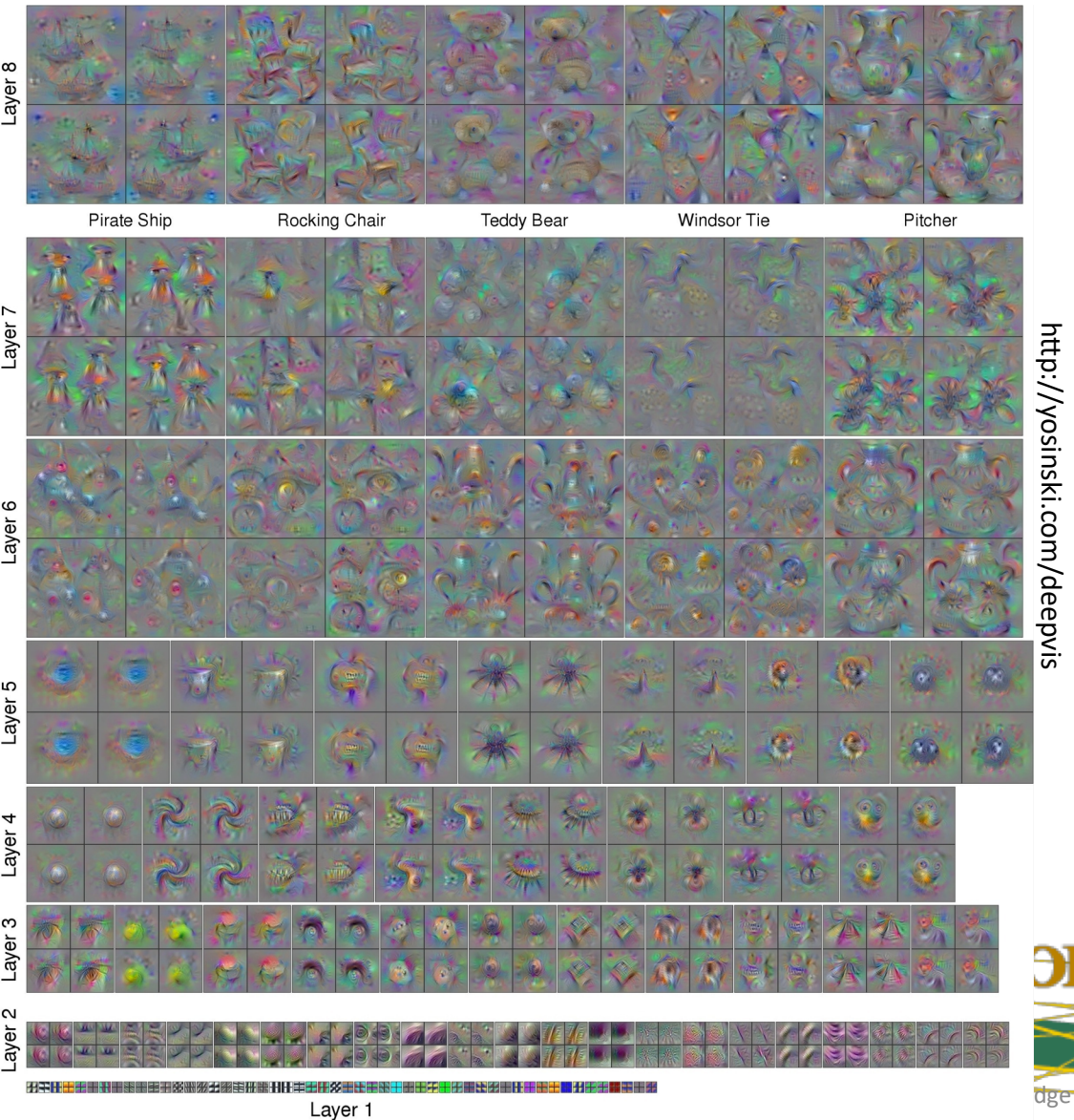
Conv 1: Edge+Blob | **Conv 3: Texture** | **Conv 5: Object Parts** | **Fc8: Object Classes**

http://vision03.csail.mit.edu/cnn_art/

Analyzing a CNN

- Performance tells you *how well* it learns
- Analysis tells you *what* it learns
- Analysis methods still under development
- For any node n:
  - Identify the training samples generating the highest activations
  - Compute $\partial C/\partial n$, use gradient ascent to create maximal activation image
    - Images do not look "real"
    - Add additional constraints (like minimizing L2) to create smooth inputs

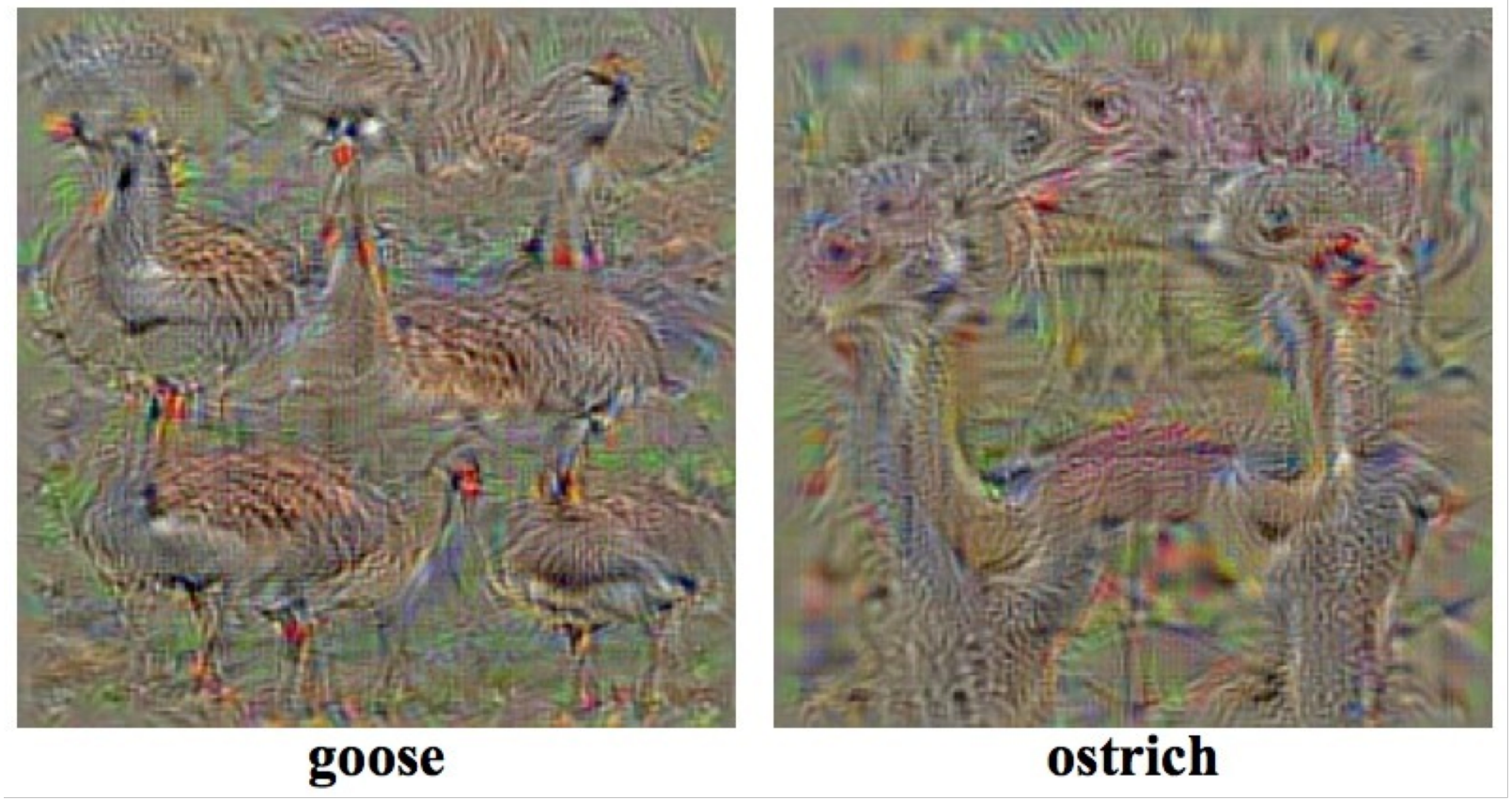
Gradient-ascent Analysis



- Images generated by using partial derivatives to create images that maximize activations
- These images also use L2 regularization to avoid images that look like white noise

Final Layer Features

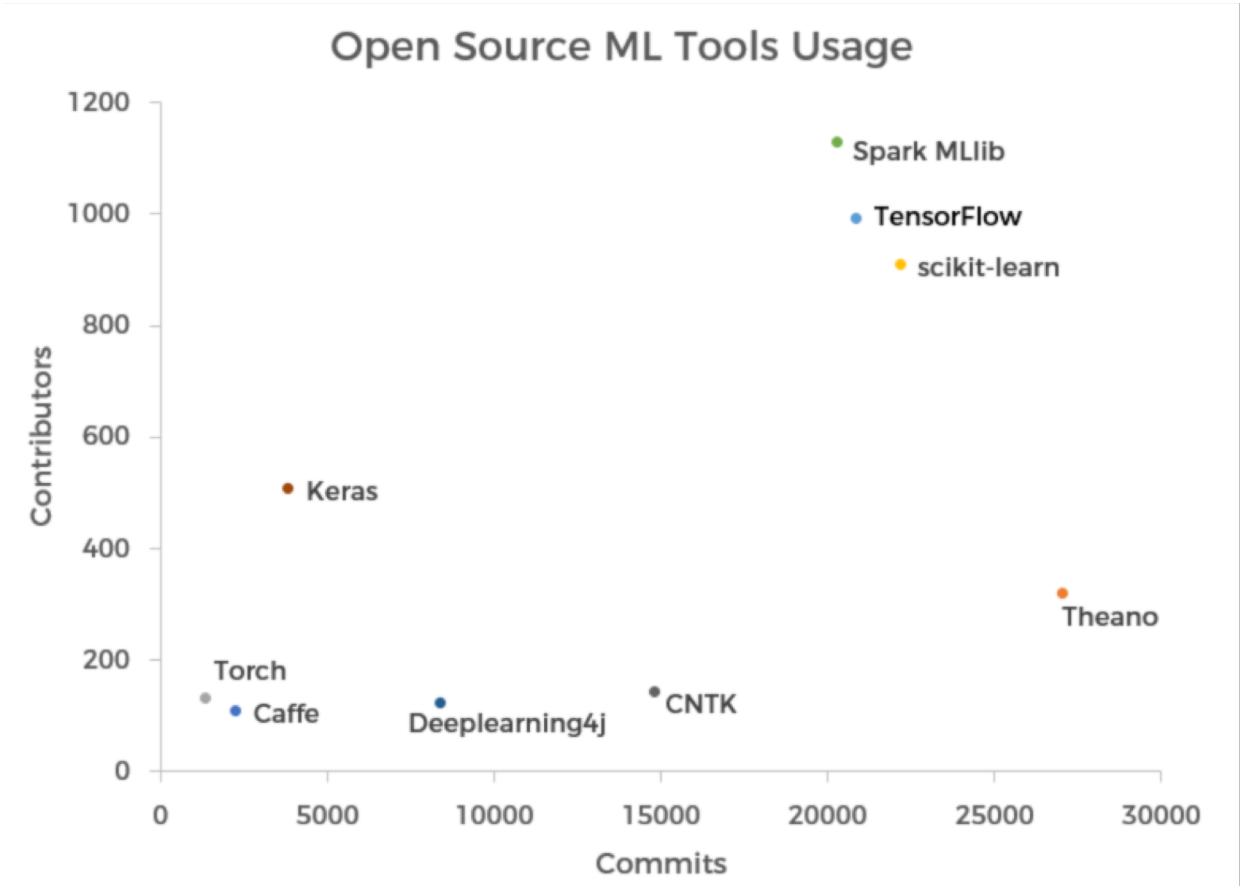
http://yosinski.com/static/proj/deepvis_goose_ostrich.jpg



Created with gradient-ascent optimization and L2 regularization

GETTING STARTED WITH TENSORFLOW

Step Up a Level …



Graphic from altexsoft.

*https://www.altexsoft.com/blog/datascience/choosing-an-open-source-machine-learning-framework-tensorflow-theano-torch-scikit-learn-caffe/

Think Locally

and historically

Cameron: High Level Language Compilation for Reconfigurable Systems

Jeff Hammes, Bob Rinker, Wim Bo¨hm, Walid Najjar, Bruce Draper, Ross Beveridge

Department of Computer Science, Colorado State University

1999

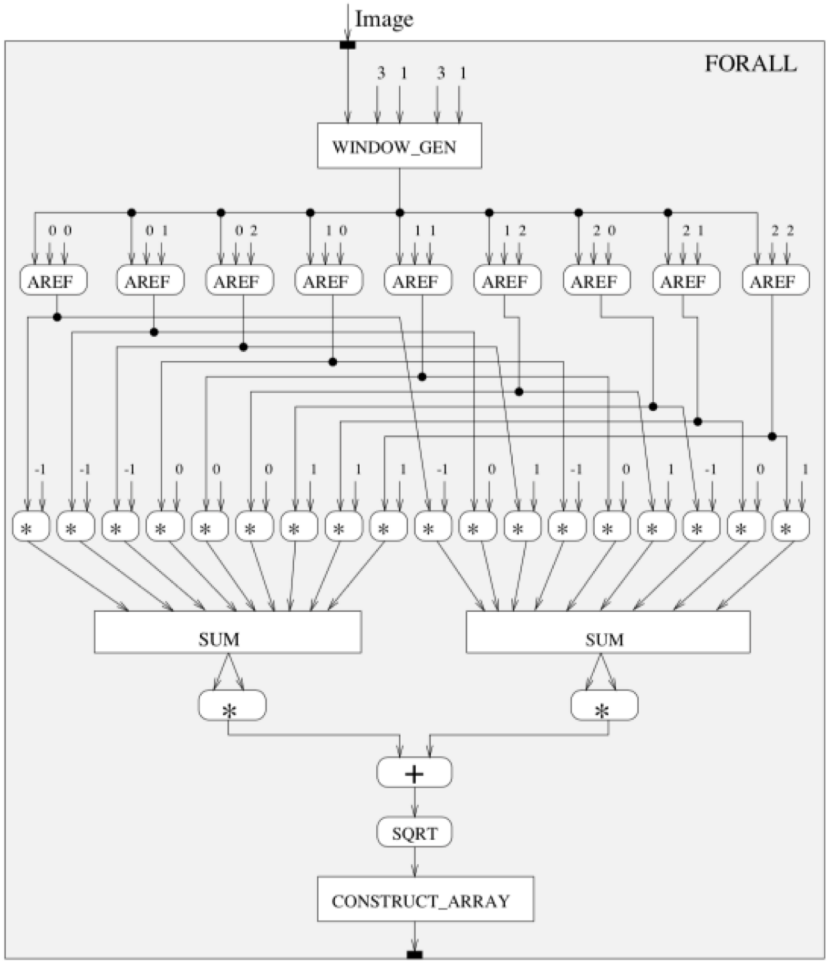


Figure 4. DDCF graph for Prewitt program after loop unrolling and array value propagation.

Tensorflow Learning Resources

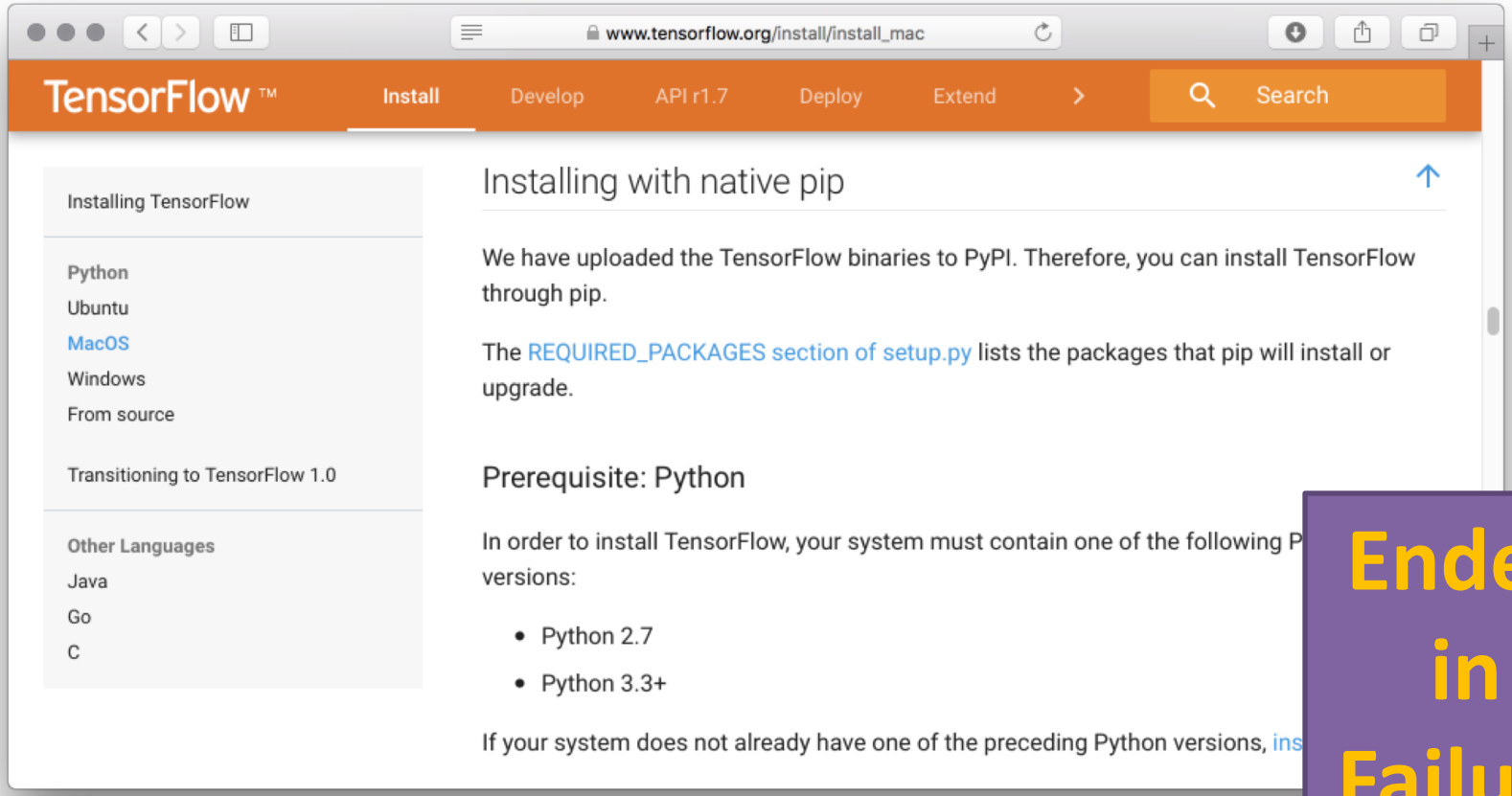
- First if course, tensorflow.org …
- Then …

Tensorflow

The web contains many helpful tutorials on tensorflow. I have only begun to scratch the surface. That caveat offered, I found these helpful.

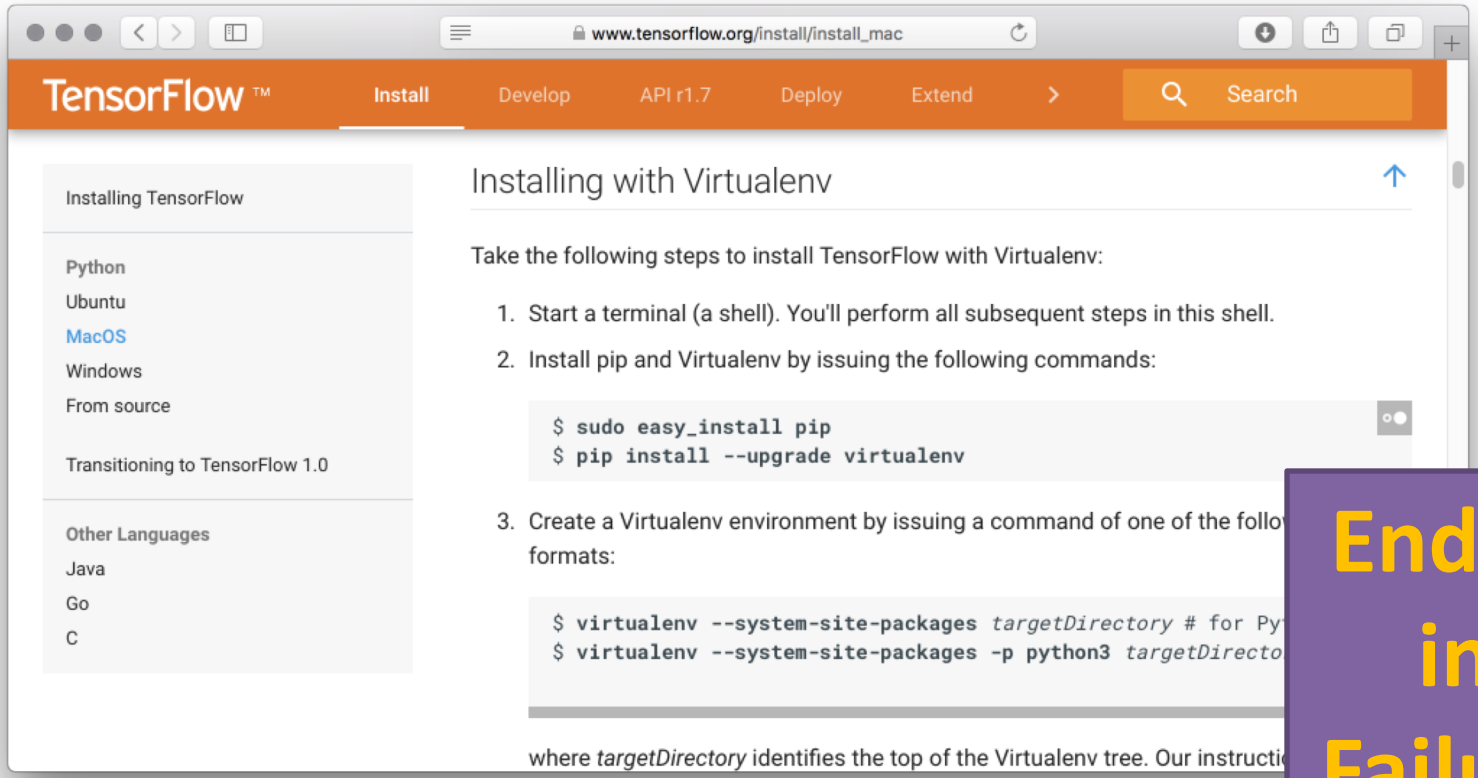
- TensorFlow 101 (Really Awesome Intro Into TensorFlow). This is a relatively long but I thought really excellent walk through to set the context for Tensorflow.
- TensorFlow Tutorial - How to use TensorFlow to Build a Neural Network. This is relatively shorter - just under 8 minutes - introduction with an emphasis on image recognition.
- Dan Aloni's blog post on Back Propagation with TensorFlow. We will use this tutorial to begin digging into the basics of using Tensorflow.

Nitty Gritty – Installing 1



Ended in Failure

Nitty Gritty – Installing 2



Ended in Failure

Nitty Gritty – Installing 3

Ended in Success

Recommended by Asa Ben-Hur!

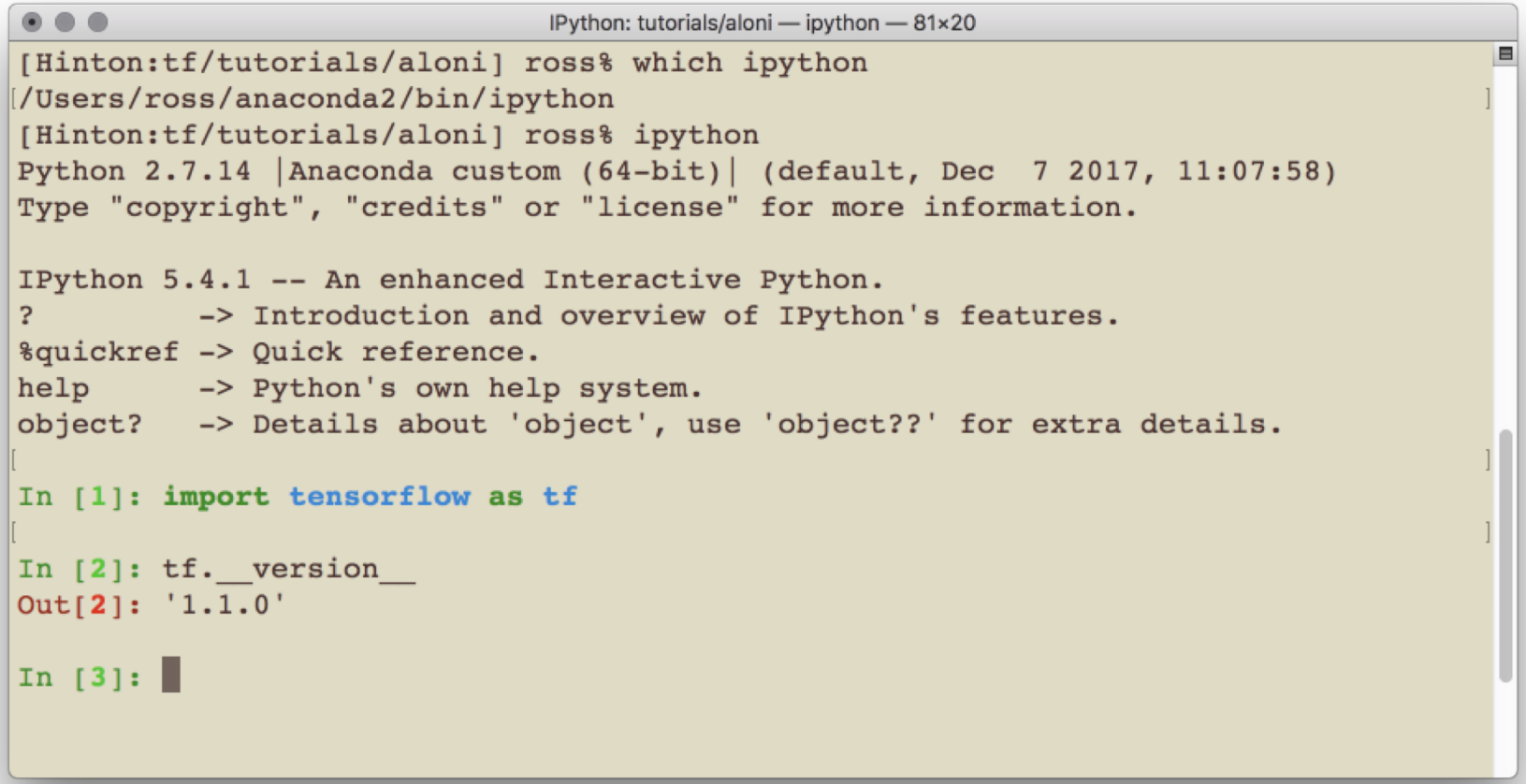
Using Conda

- First, install
- Next, update your unix path

```
[Hinton:tf/tutorials/aloni] ross% which conda
/Users/ross/anaconda2/bin/conda
```

- Now review basic commands
  - update
  - install
  - list

More Basics - iPython



Add two (constant) Numbers

- From documentation on tf.Session
- Start thinking about a data flow graph

```
# Build a graph.
a = tf.constant(5.0)
b = tf.constant(6.0)
c = a * b

# Launch the graph in a session.
sess = tf.Session()

# Evaluate the tensor `c`.
print(sess.run(c))
```

Add two Variables

- This is a bit more tricky
- The global variable initializer is important!

```
In [3]: a = tf.Variable(3.0)

In [4]: b = tf.Variable(4.0)

In [5]: c = a + b

In [6]: init_op = tf.global_variables_initializer()

In [7]: sess.run(init_op)

In [8]: sess.run(c)
Out[8]: 7.0
```