Brief Introduction to Deep Learning and TensorFlow

Deep Learning in Earth Science
Lecture 1
By Xiao Zhuowei

For researchers interested in studying Earth science with deep learning.

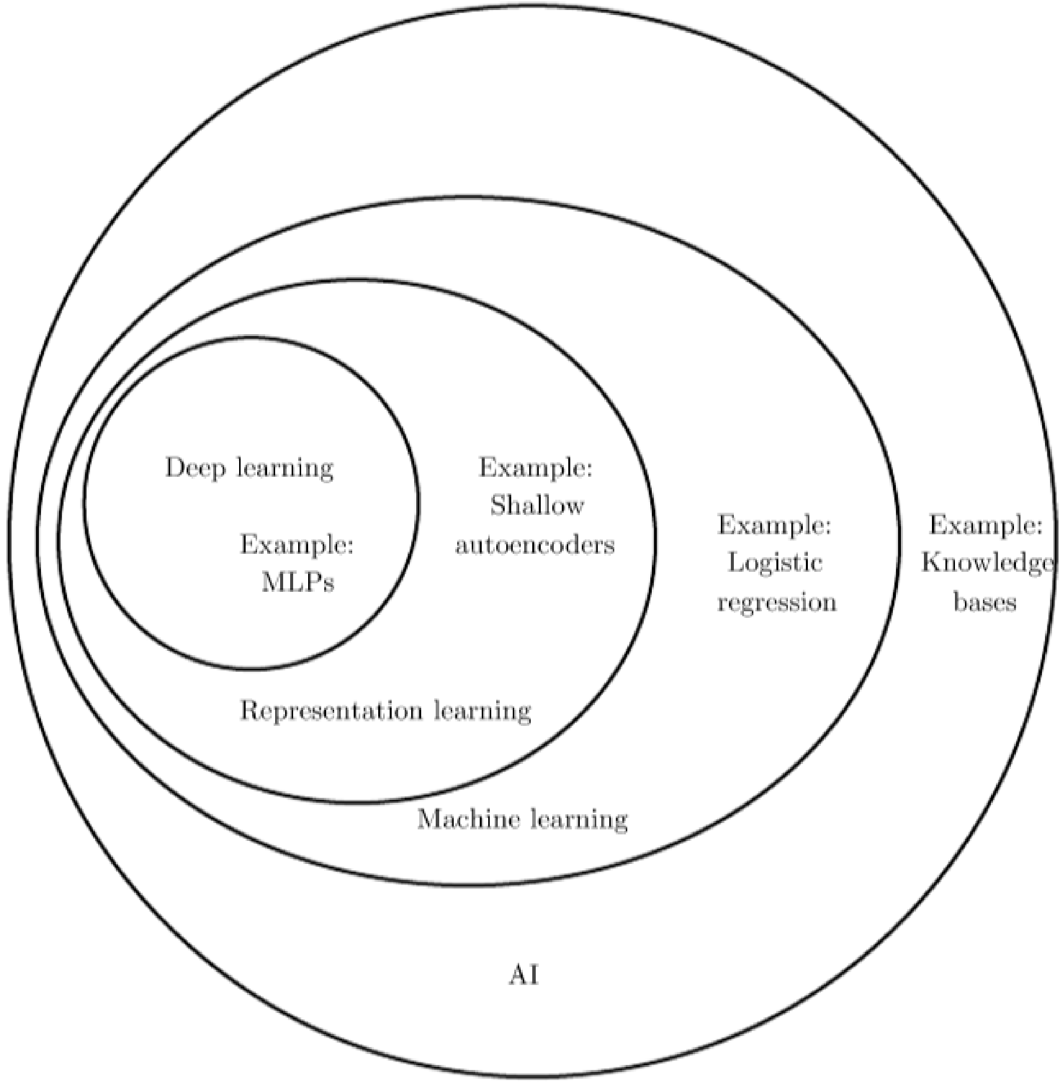
All resources in lectures are available at https://github.com/MrXiaoXiao/DLiES

OUTLINES

1. Brief Introduction to Deep Learning
2. TensorFlow Basics
3. Classifying Stability of Mantle with Neural Networks: An Example
4. Discussions

Brief Introduction to Deep Learning

Deep Learning is about automatically obtaining ***<u>representation of input</u>*** and ***<u>mapping (from representation) to output</u>*** with deep neural network architectures.



(Deep Learning, MIT Press, 2016)

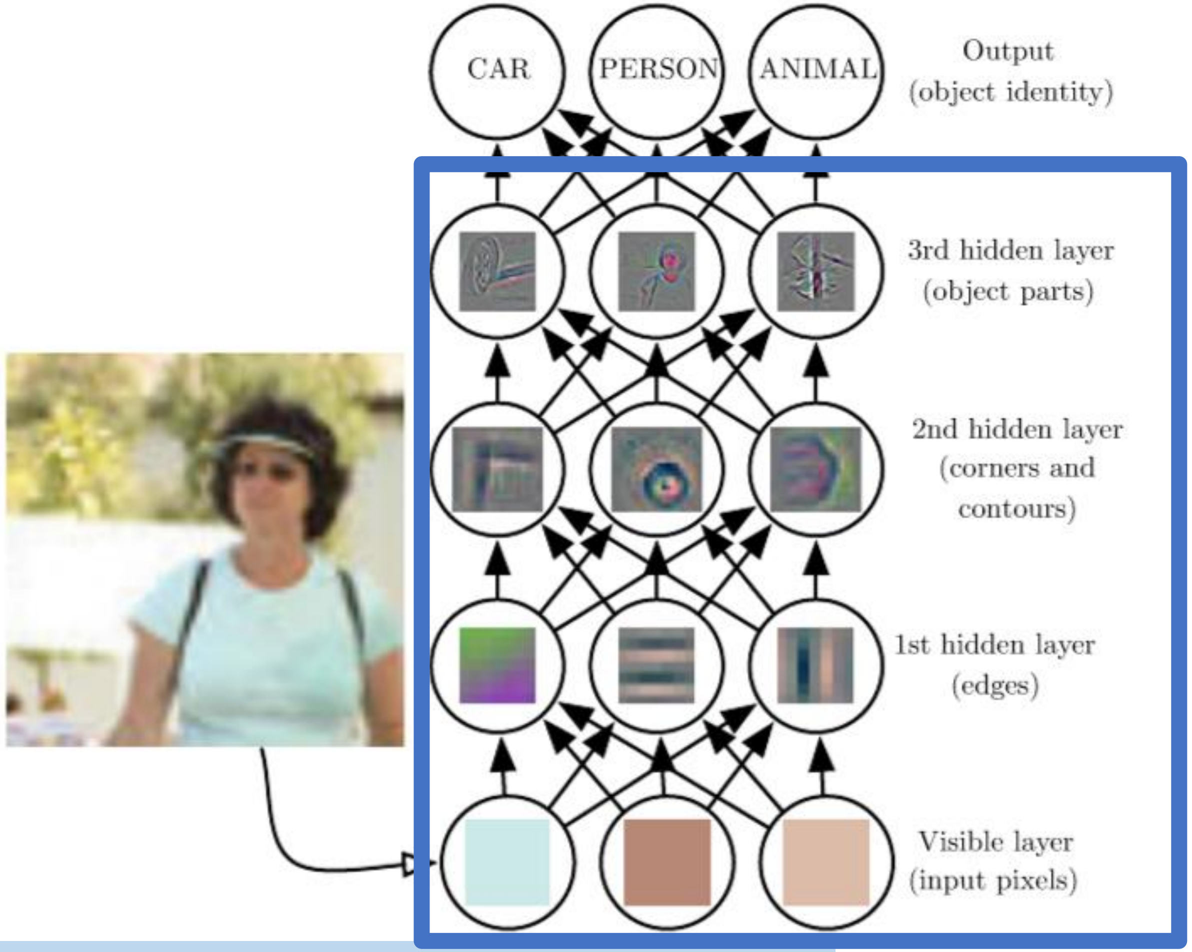
Brief Introduction to Deep Learning

Obtain the ***representation*** of input.

What We See

What Computers See

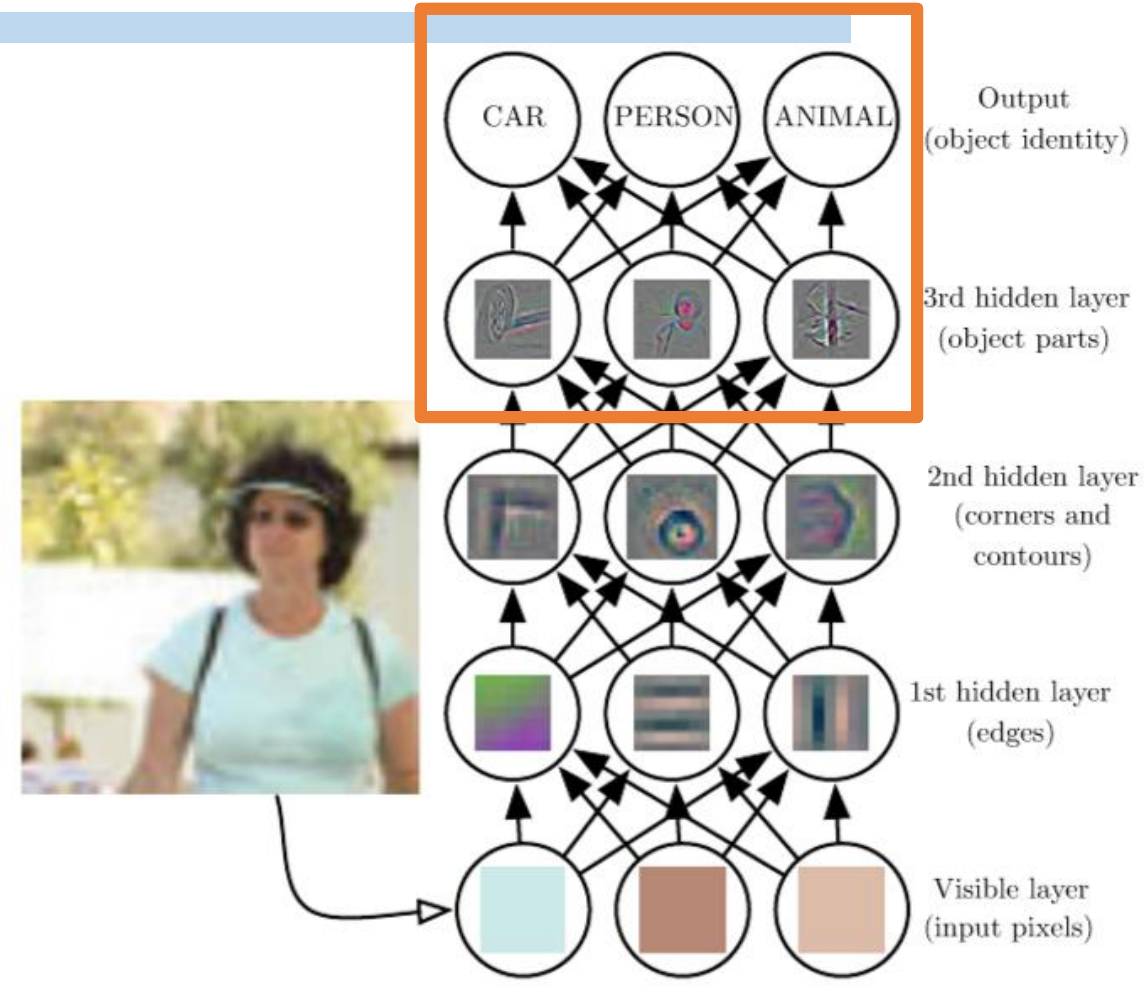
(https://adeshpande3.github.io)



(Deep Learning, MIT Press, 2016)

Brief Introduction to Deep Learning

Obtain ***mapping*** from representation to output.

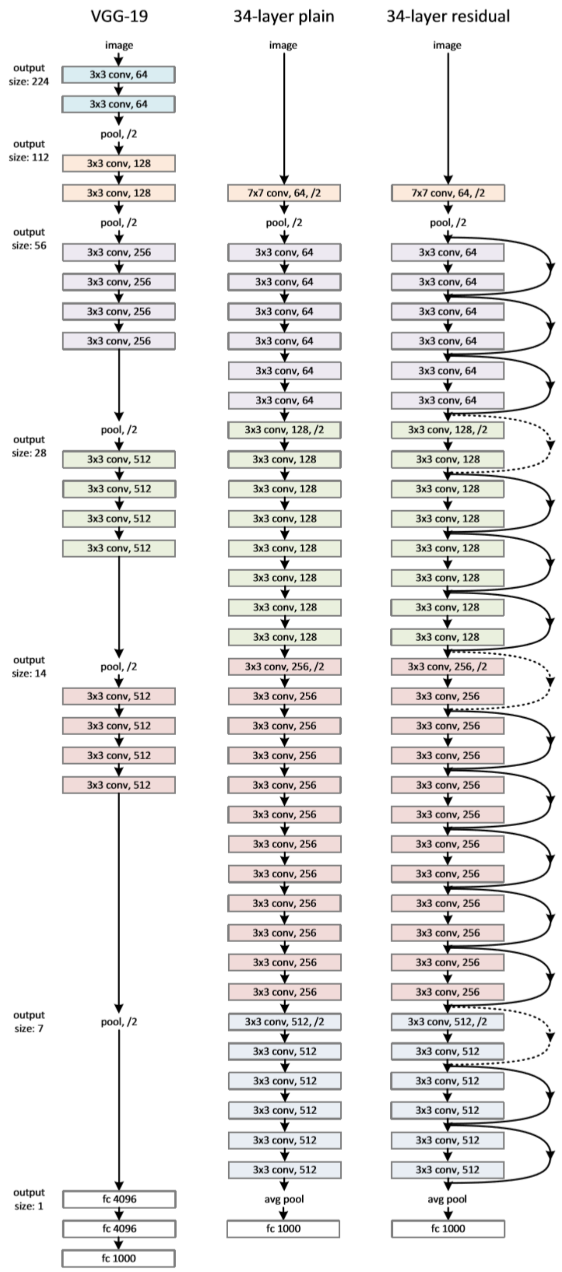


(Deep Learning, MIT Press, 2016)

Brief Introduction to Deep Learning

Complicated representations are built out of simpler ones.

The graph of deep learning architecture is deep, with many layers.



Example network architectures (He et al., 2015)

Brief Introduction to Deep Learning

Considering deep learning as algorithm for non-linear function approximation

$$Ideal\ Output = Ideal\ Function(Input + Noise)$$

$$Approximation\ of\ Ideal\ Output = DL\ Model(Input + Noise)$$

What can deep learning do in Earth science?

Classification

Denoising

Forward Modeling

Inversion

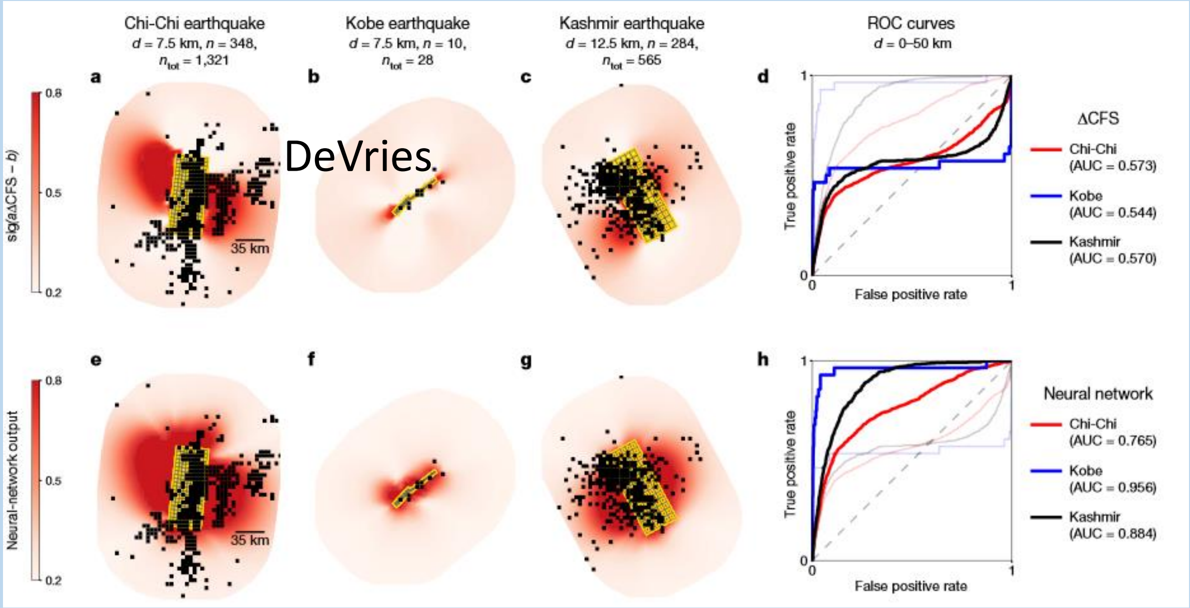
…

Brief Introduction to Deep Learning

What can deep learning do in Earth science?

Classification

Predicting aftershocks following large earthquakes

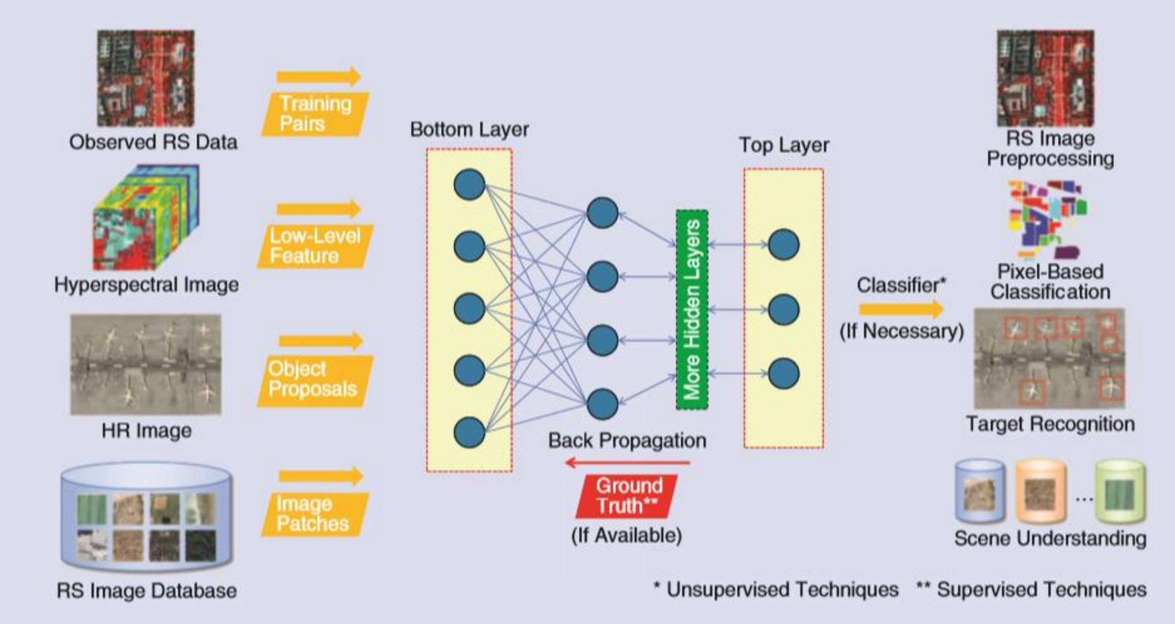


(DeVries et al., 2018)

What can deep learning do in Earth science?

Classification

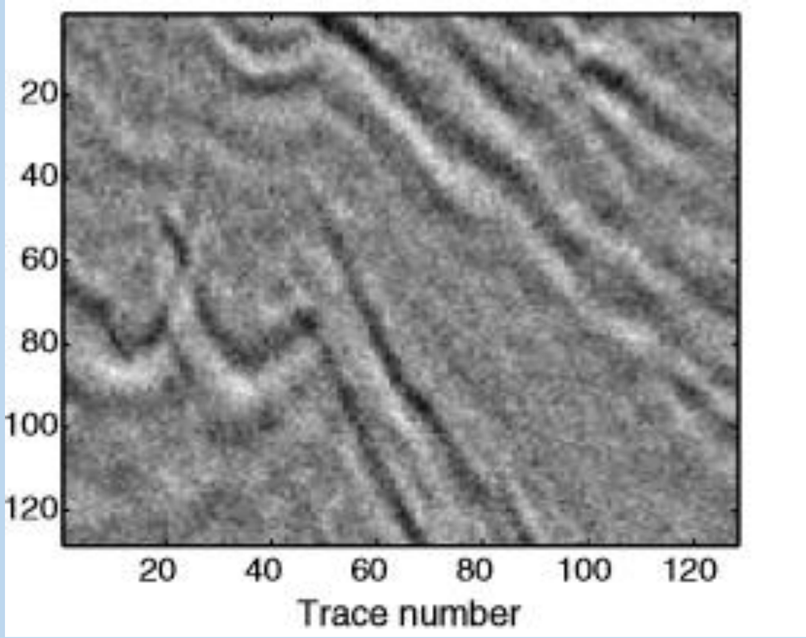
Processing remote sensing data



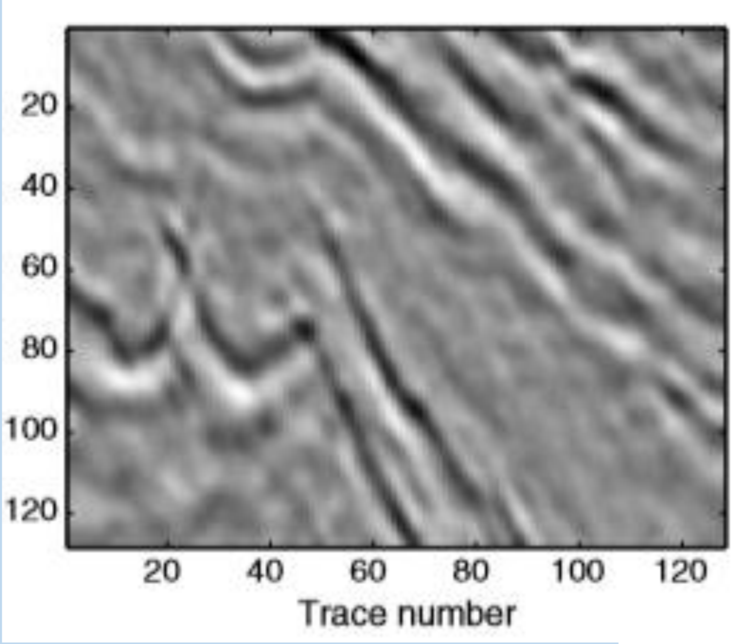
(Zhang et al., 2016)

What can deep learning do in Earth science?

Denoising



Noisy input



DL output

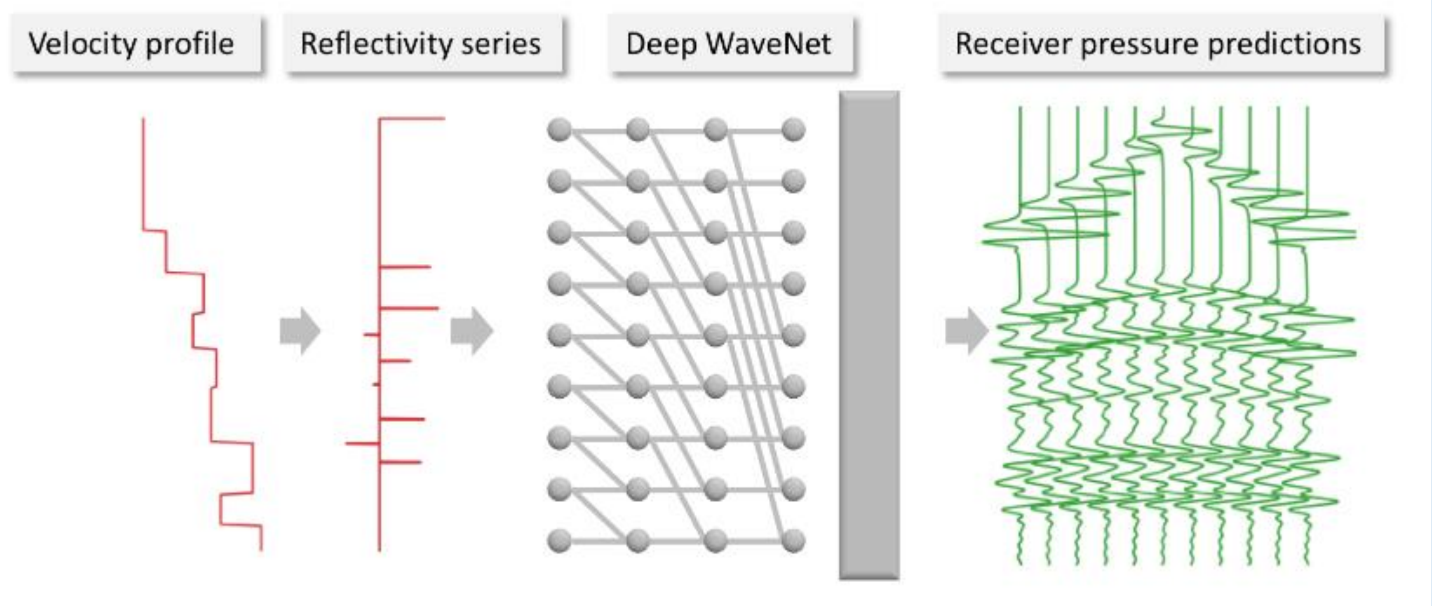
(Beckouche and Ma, 2014)

Brief Introduction to Deep Learning

What can deep learning do in Earth science?

Forward Modeling

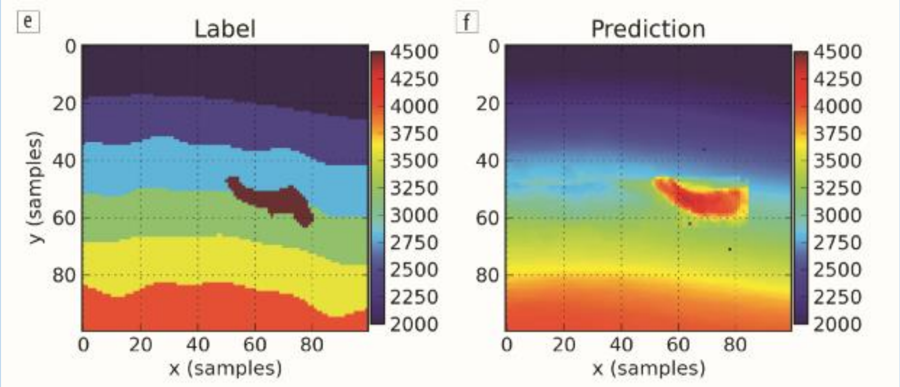
Fast approximate simulation of seismic waves with deep learning



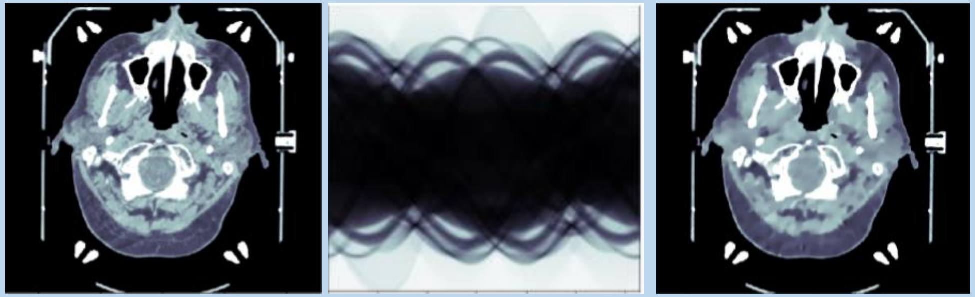
(Moseley et al., 2018)

What can deep learning do in Earth science?

Inversion



(Araya-Polo et al., 2018)



Model Observation Inversion by DL

(Adler and Öktem, 2017)

OUTLINES

1. Brief Introduction to Deep Learning
2. TensorFlow Basics
3. Classifying Stability of Mantle with Neural Networks: An Example
4. Discussions

TensorFlow Basics

TensorFlow™ is an open source software library for high performance numerical computation.

https://www.tensorflow.org/

or

https://tensorflow.google.cn/

Install TensorFlow via Anaconda

Anaconda Distribution is a free, easy-to-install package manager, environment manager and Python distribution with a collection of 1,000+ open source packages with free community support.

Anaconda Download (https://www.anaconda.com/download/)

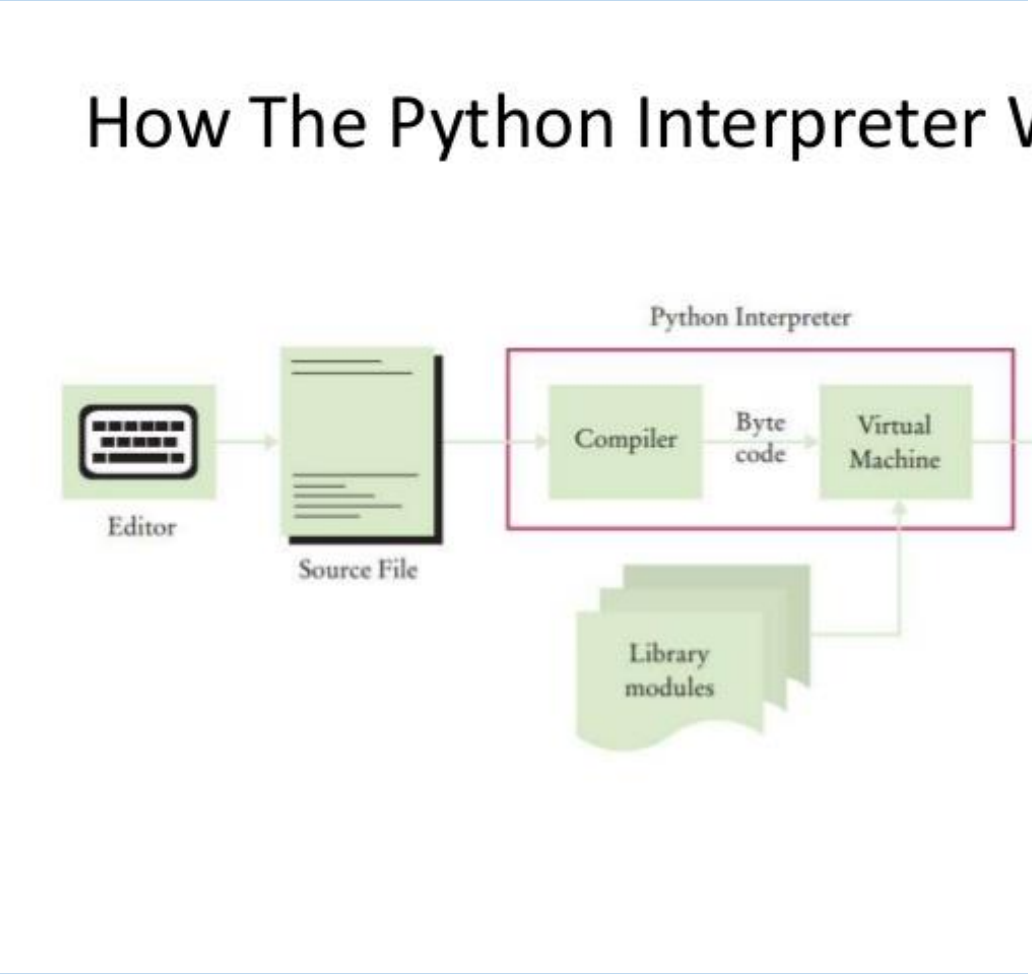
Tensorflow-in-Anaconda
(https://www.anaconda.com/blog/developer-blog/tensorflow-in-anaconda/)

TensorFlow Basics

Python is an interpreted high-level programming language for general-purpose programming.

python™

(https://www.python.org/)



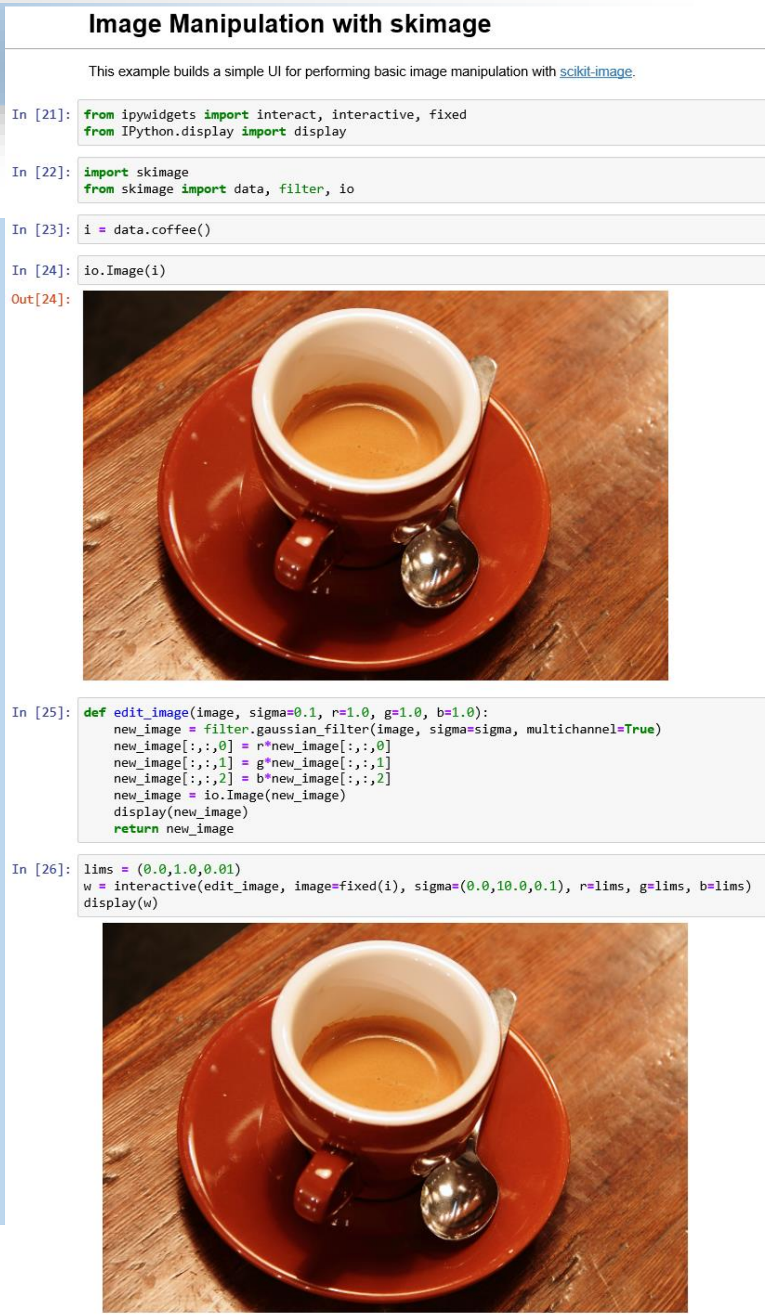
(http://opensourceforgeeks.blogspot.com/2015/10/how-python-works.html)

TensorFlow Basics

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text.

https://jupyter.org/

jupyter



TensorFlow Basics

TensorFlow Hello World

TensorFlow Hello World

Modified from https://github.com/aymericdamien/TensorFlow-Examples/blob/master/examples/1_Introduction/helloworld.py

In [1]:
```
import tensorflow as tf
```

In [2]:
```
# Simple hello world using TensorFlow

# Create a Constant op

# The op is added as a node to the default graph.

#

# The value returned by the constructor represents the output

# of the Constant op.

hello = tf.constant('Hello, TensorFlow!')
```

In [3]:
```
print(hello)
```

```
Tensor("Const:0", shape=(), dtype=string)
```

In [4]:
```
# Start tf session

sess = tf.Session()
```

In [5]:
```
# Run the op

print(sess.run(hello))
```

```
b'Hello, TensorFlow!'
```

TensorFlow Multiply Matrices

TensorFlow Multiply Matrices Example

Modified from https://github.com/vahidk/EffectiveTensorflow

In [1]:
```
import tensorflow as tf
```

In [2]:
```
x = tf.random_normal([3,3])
y = tf.random_normal([3,3])
z = tf.matmul(x,y)
```

In [3]:
```
print('{}\n{}\n{}'.format(x,y,z))
```

```
Tensor("random_normal:0", shape=(3, 3), dtype=float32)
Tensor("random_normal_1:0", shape=(3, 3), dtype=float32)
Tensor("MatMul:0", shape=(3, 3), dtype=float32)
```

In [4]:
```
sess = tf.Session()
z_val = sess.run(z)
```

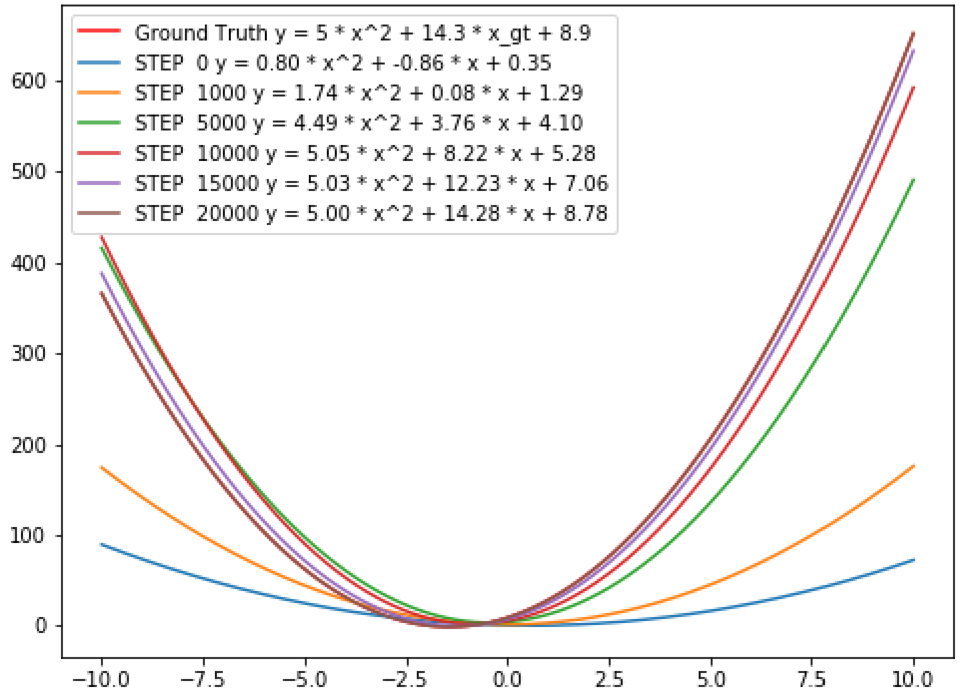
In [5]:
```
print(z_val)
```

```
[[-1.8857789   0.02845232  2.23009   ]
 [ 0.20160252  0.49441913  0.37605742]
 [ 3.5984905   1.7590961  -0.84973013]]
```

TensorFlow Basics

Approximate Quadratic Function With TensorFlow

```python
plt.figure(figsize=(8,6))

x_axis = np.arange(-10.0,10.0,0.0001)
y_gt = 5.0 * np.square(x_axis) + 14.3 * x_axis + 8.9
plt.plot(x_axis,y_gt,color='r',label='Ground Truth y = 5 * x^2 + 14.3 * x_gt + 8.9')
for data in inspect_list:
    yhat = data['w'][0][0] * np.square(x_axis) + data['w'][1][0] * x_axis + data['w'][2][0]
    plt.plot(x_axis,yhat,label='STEP  {:} y = {:.2f} * x^2 + {:.2f} * x + {:.2f}'.format(
        data['step'],data['w'][0][0],data['w'][1][0],data['w'][2][0]))
#plt.xlim([-10.0,10.0])
plt.legend()
plt.show()
```



Recommended TensorFlow tutorials

Effective TensorFlow
https://github.com/vahidk/EffectiveTensorflow

TensorFlow Official Tutorial
https://www.tensorflow.org/tutorials/

Simple and ready-to-use tutorials for TensorFlow
https://github.com/astorfi/TensorFlow-World

OUTLINES

1 Brief Introduction to Deep Learning

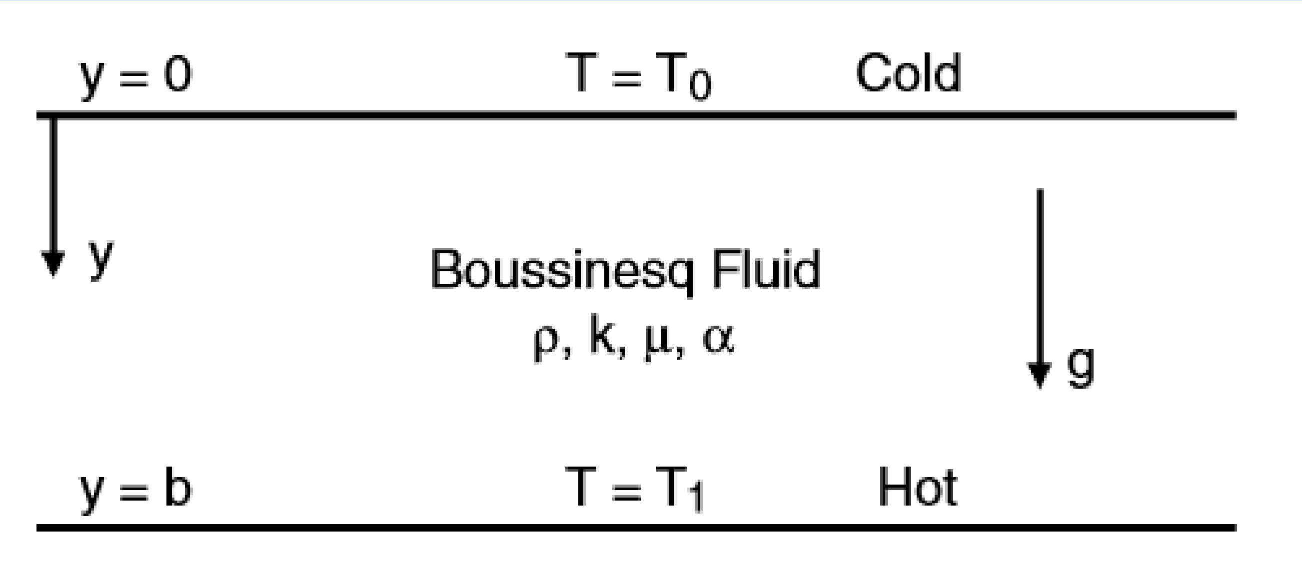
2 TensorFlow Basics

3 Classifying Stability of Mantle with Neural Networks: An Example

4 Discussions

Classifying Stability of Mantle with Neural Networks: An Example

Plane Layer Heated from Below



(Schuber et al., 2001)

Full Connection and Back Propagation

Classifying Stability of Mantle with Neural Networks: An Example

```
#Define a function to generate data set
def generate_data_set(instance_num = 10000, split_rate = 0.6):
    #instance[0]  Gravitational acceleration
    #instance[1]  Volume expansion coefficient
    #instance[2]  Kinematic viscosity coefficient
    #instance[3]  Thermal diffusivity
    #instance[4]  Depth
    #instance[5]  b
    #instance[6]  λ
    #instance[7]  (T0 - T1)/1000
    #instance[8] stability 0 is unstable and 1 is stable
    data_set = np.zeros([instance_num, 9])

    #simulate gravitational accelerations
    data_set[:,0] = np.random.uniform(8,10,size=instance_num)
    #simulate Volume expansion coefficient
    data_set[:,1] = np.random.uniform(1e-4,1e-2,size=instance_num)
    #simulate Kinematic viscosity coefficient
    data_set[:,2] = np.random.uniform(1e-6,1e-2,size=instance_num)
    #simulate Thermal diffusivity
    data_set[:,3] = np.random.uniform(1.0,10.0,size=instance_num)
    #simulate Depth 1000km
    data_set[:,4] = np.random.uniform(0,3.5,size=instance_num)
    for idx in range(instance_num):
        #simulate b 1000km
        data_set[idx,5] = np.random.uniform(max([2.5,data_set[idx,4]]),3.5)
    #simulate  λ
    data_set[:,6] = np.random.uniform(0.0,6.0,size=instance_num)
    #simulate T0 - T1
    data_set[:,7] = np.random.uniform(0.0,5.0,size=instance_num)
    #simulate stability
    for idx in range(instance_num):
        #
        Ra = (data_set[idx,0]*data_set[idx,1]* data_set[idx,7]*1000
              *(data_set[idx,4])**3)/(data_set[idx,2]*data_set[idx,3])
        Racr = (((np.pi**4)*((4.0+(data_set[idx,6]/data_set[idx,5])**2)**3))
                /(4*((data_set[idx,6]/data_set[idx,5])**4)))
        if Ra > Racr:
            data_set[idx,8] = 0
        else:
            data_set[idx,8] = 1
    split_index = int(instance_num*split_rate)

    train_set = data_set[0:split_index,:]
    test_set = data_set[1:split_index,:]

    return train_set,test_set
```

Classifying Stability of Mantle with Neural Networks: An Example

```
In [4]: data_set_size = 1000000
        split_rate = 0.5
        train_set, test_set = generate_data_set(instance_num = data_set_size, split_rate = split_rate)

Stable:574480 UnStable:425520
```

```
In [5]: #define full connection layer
        def full_connection_layer(input_tensor, n_out,
                                  w_init=tf.truncated_normal_initializer(stddev=0.1),
                                  b_init=tf.constant_initializer(0.1),
                                  activation=tf.nn.sigmoid, name=None):
            n_in = input_tensor.get_shape().as_list()[1]
            with tf.variable_scope(name):
                weight = tf.get_variable('weight', [n_in, n_out], initializer=w_init)
                bias = tf.get_variable('bias', [n_out], initializer=b_init)
            output_tensor = activation(tf.matmul(input_tensor, weight)+bias, name=name+'_output')
            return output_tensor
```

```
In [6]: def inference(input_tensor):
            hidden_layer_1 = full_connection_layer(input_tensor=input_tensor, n_out=4, name='fc_layer_1')
            hidden_layer_2 = full_connection_layer(input_tensor=hidden_layer_1, n_out=4, name='fc_layer_2')
            hidden_layer_3 = full_connection_layer(input_tensor=hidden_layer_2, n_out=4, name='fc_layer_3')
            pred = full_connection_layer(input_tensor=hidden_layer_3, n_out=2, name='pred')
            return pred
```

```
In [7]: #set param for training
        step_num = 20001
        batch_size = 1000
        data_length = 8
        learning_rate = 0.01
        #setup training
        input_tensor = tf.placeholder(tf.float32, [None, data_length], name='input')
        label = tf.placeholder(tf.float32, [None, 2], name='label')
        pred = inference(input_tensor=input_tensor)
        loss = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(logits=pred, labels=label))
        train_op = tf.train.AdamOptimizer(learning_rate = learning_rate).minimize(loss)
```

```
In [8]: def get_traning_batch(train_set, batch_size, data_length):
            batch_ids = np.random.choice(len(train_set['input']), batch_size)
            input_batch = np.zeros([batch_size, data_length])
            label_batch = np.zeros([batch_size, 2])
            for idx in range(batch_size):
                input_batch[idx][:] = train_set['input'][batch_ids[idx]][:]
                label_batch[idx][:] = train_set['label'][batch_ids[idx]][:]
            return input_batch, label_batch
```

```
In [9]: sess = tf.Session()
        sess.run(tf.global_variables_initializer())
```

```
In [10]: #start traning
         for idx in range(step_num):
             input_batch, label_batch = get_traning_batch(train_set, batch_size, data_length)
             _, loss_val = sess.run([train_op, loss], {input_tensor: input_batch, label: label_batch})
             if idx%2000 == 0:
                 print(loss_val)

0.6934938
0.39525732
0.3629702
0.3573489
0.35604522
0.34598345
0.34709588
0.34188947
0.354488
0.3468753
0.3486143
```

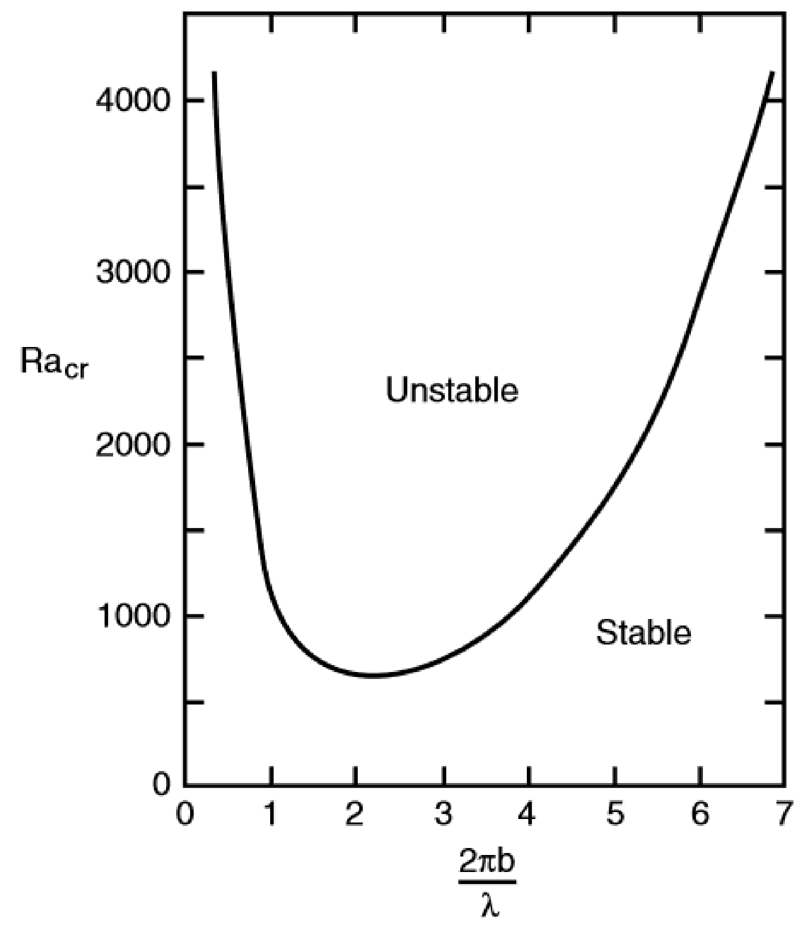
```
In [11]: correct_pred = tf.equal(tf.argmax(pred, 1), tf.argmax(label, 1))
         accuracy = tf.reduce_mean(tf.cast(correct_pred, tf.float32))
         print('Accuracy: {}'.format(sess.run(accuracy, {input_tensor: test_set['input'], label: test_set['label']})))

Accuracy: 0.9649959802627563
```

Classifying Stability of Mantle with Neural Networks: An Example

$$Ra = Ra_{cr} = \frac{\left(\pi^2 + 4\pi^2/\lambda^{*2}\right)^3}{4\pi^2/\lambda^{*2}} = \frac{\pi^4}{4\lambda^{*4}}\left(4 + \lambda^{*2}\right)^3$$

$$Ra = \frac{\alpha g \left(T_1 - T_0\right) b^3}{\nu\kappa}$$

$$\lambda^* = \lambda/b$$



(Schuber et al., 2001)

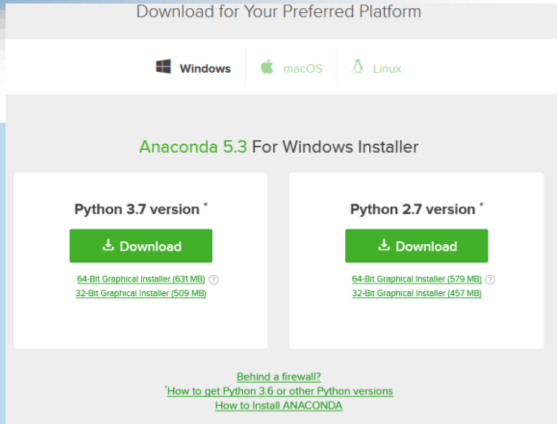
Classifying Stability of Mantle with Neural Networks: An Example

OUTLINES

1. Brief Introduction to Deep Learning
2. TensorFlow Basics
3. Classifying Stability of Mantle with Neural Networks: An Example
4. Discussions

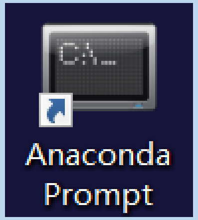
Discussions

TensorFlow Installation via Anaconda

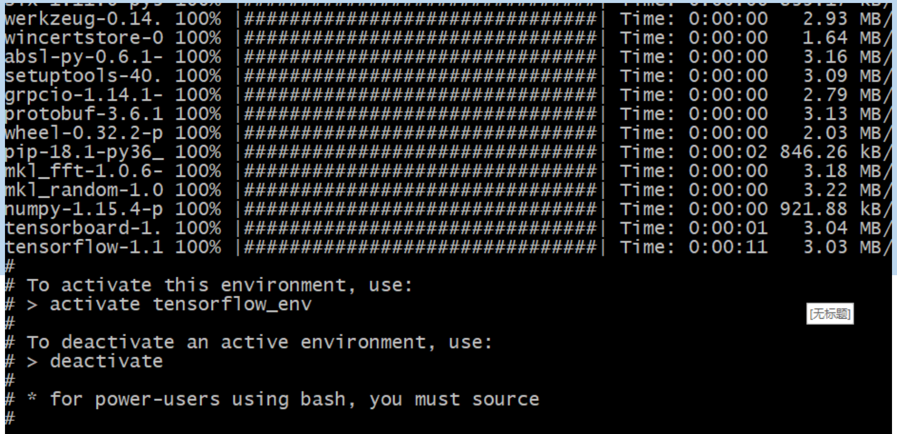


Step 1. Install Anaconda from (https://www.anaconda.com/download/)

Step 2. Create a new conda environment containing TensorFlow.
Open Anaconda Prompt and run
'conda create -n tensorflow_env tensorflow python=3.6'
or
'conda create -n tensorflow_gpuenv tensorflow-gpu python=3.6'
for GPU version



Congratulations…



References