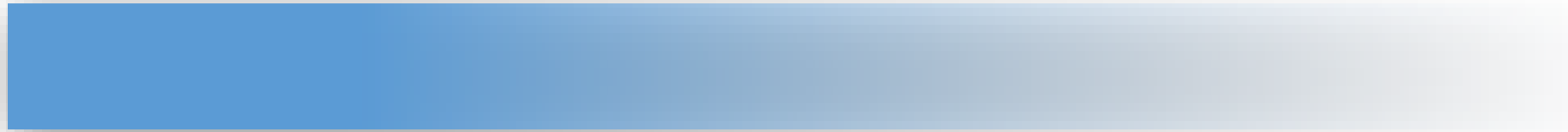




Brief Introduction to Deep Learning and TensorFlow



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

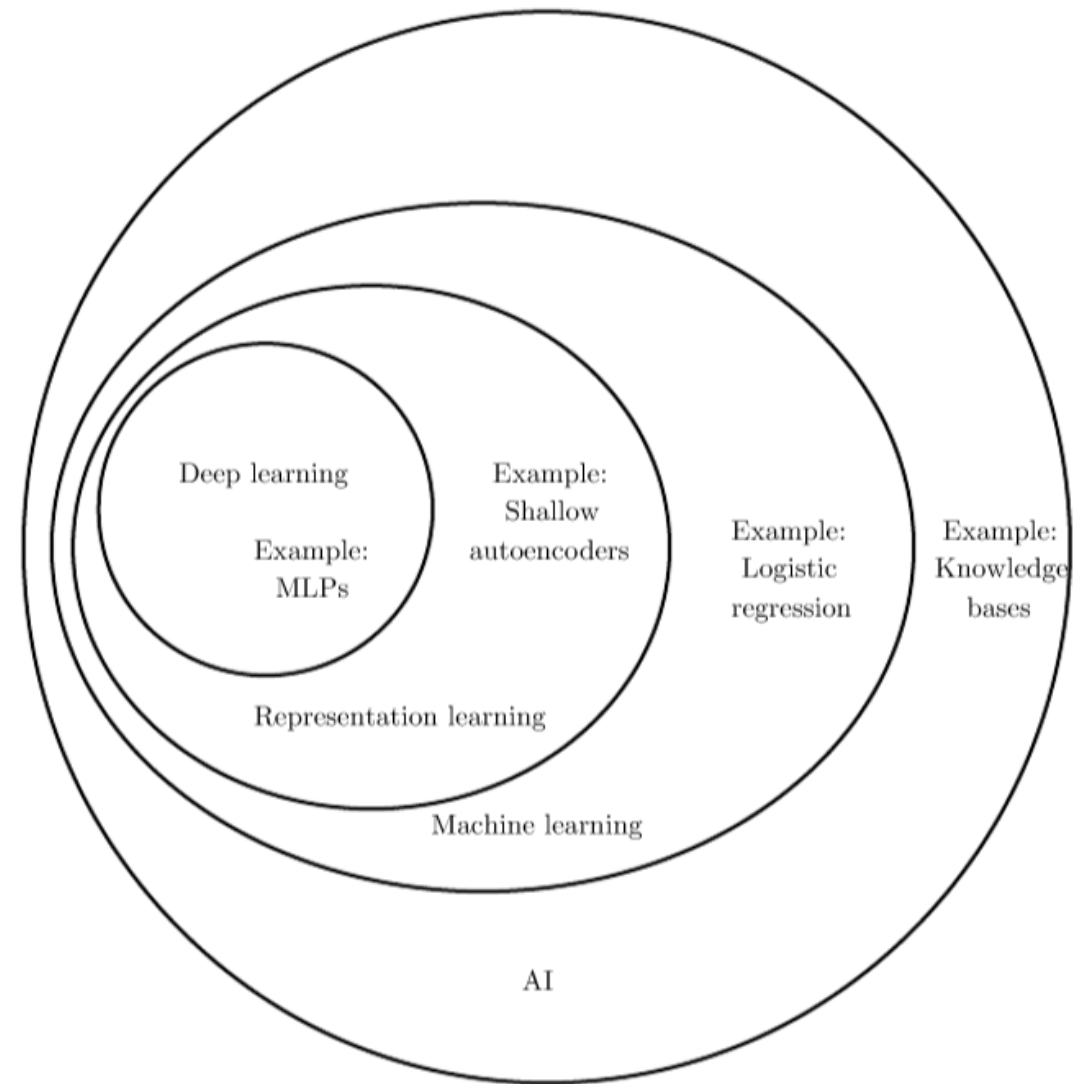
**Classifying Stability of Mantle with
Neural Networks: An Example**

4

Discussions

Brief Introduction to Deep Learning

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



Brief Introduction to Deep Learning

Obtain the *representation* of input.

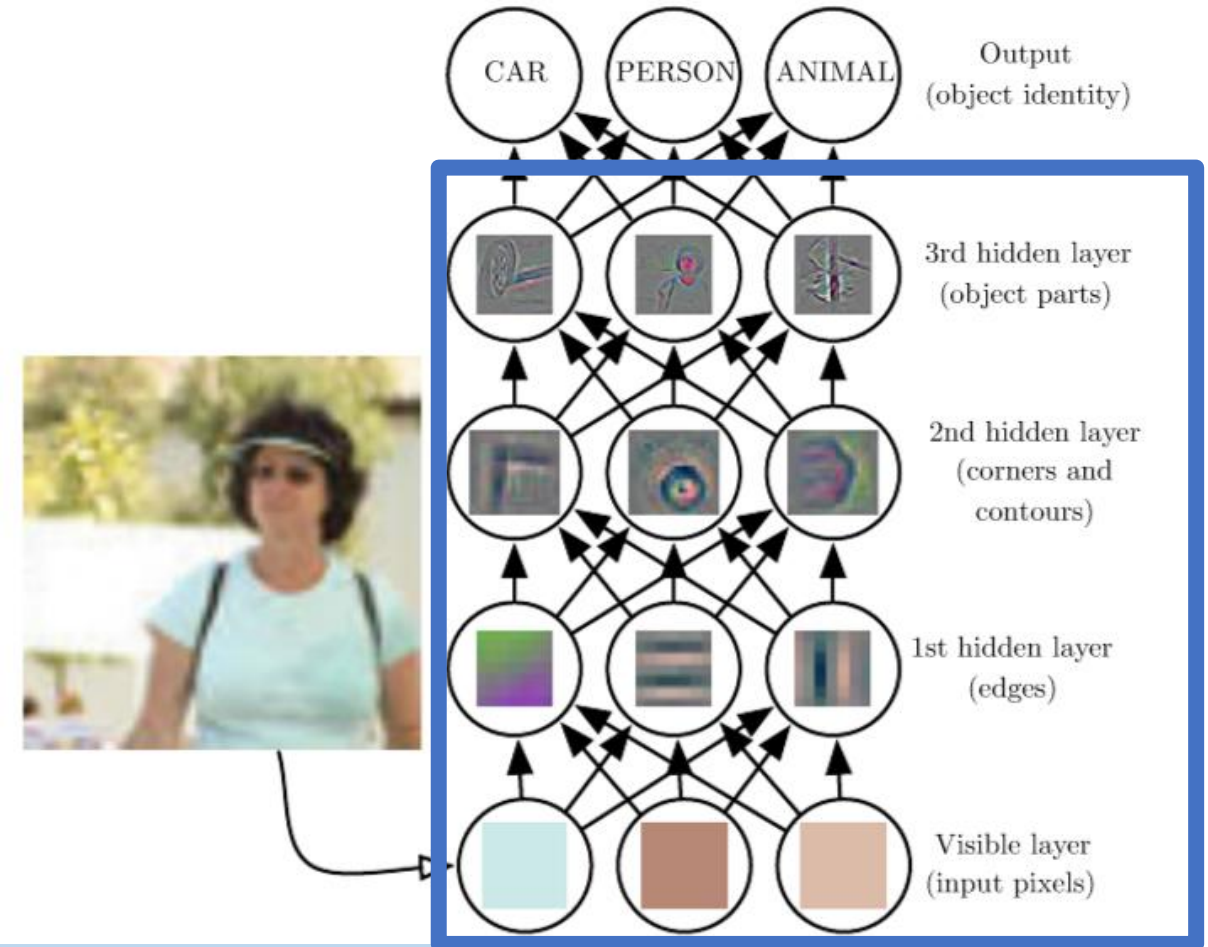


What We See

```
08 02 22 97 38 15 00 40 00 75 04 05 07 78 52 12 50 77 91 08
49 49 99 40 17 81 18 57 60 87 17 40 98 43 69 48 04 56 62 00
81 49 31 73 55 79 14 29 93 71 40 67 53 88 30 03 49 13 36 65
52 70 95 23 04 60 11 42 69 24 68 56 01 32 56 71 37 02 36 91
22 31 16 71 51 67 63 89 41 92 36 54 22 40 40 28 66 33 13 80
24 47 32 60 99 03 45 02 44 75 33 53 78 36 84 20 35 17 12 50
32 98 81 28 64 23 67 10 26 38 40 67 59 54 70 66 18 38 64 70
67 26 20 68 02 62 12 20 95 63 94 39 63 08 40 91 66 49 94 21
24 55 58 05 66 73 99 26 97 17 78 78 96 83 14 88 34 89 63 72
21 36 23 09 75 00 76 44 20 45 35 14 00 41 33 97 34 31 33 95
78 17 53 28 22 75 31 67 15 94 03 80 04 62 16 14 09 53 56 92
16 39 05 42 96 35 31 47 55 58 88 24 00 17 54 24 36 29 85 57
86 56 00 48 35 71 89 07 05 44 44 37 44 60 21 58 51 54 17 58
19 80 81 68 05 94 47 69 28 73 92 13 86 52 17 77 04 89 55 40
04 52 08 83 97 35 99 14 07 97 57 32 16 26 26 79 33 27 98 64
88 36 68 87 57 62 20 72 03 46 33 67 46 55 12 32 63 93 53 69
04 42 16 73 38 25 39 11 24 94 72 18 08 46 29 32 40 62 76 36
20 69 36 41 72 30 23 88 34 62 99 69 82 67 59 85 74 04 36 16
20 73 35 29 78 31 90 01 74 31 49 71 48 86 81 16 23 87 05 54
01 70 34 71 83 51 54 49 16 92 33 48 61 43 52 01 89 19 67 48
```

What Computers See

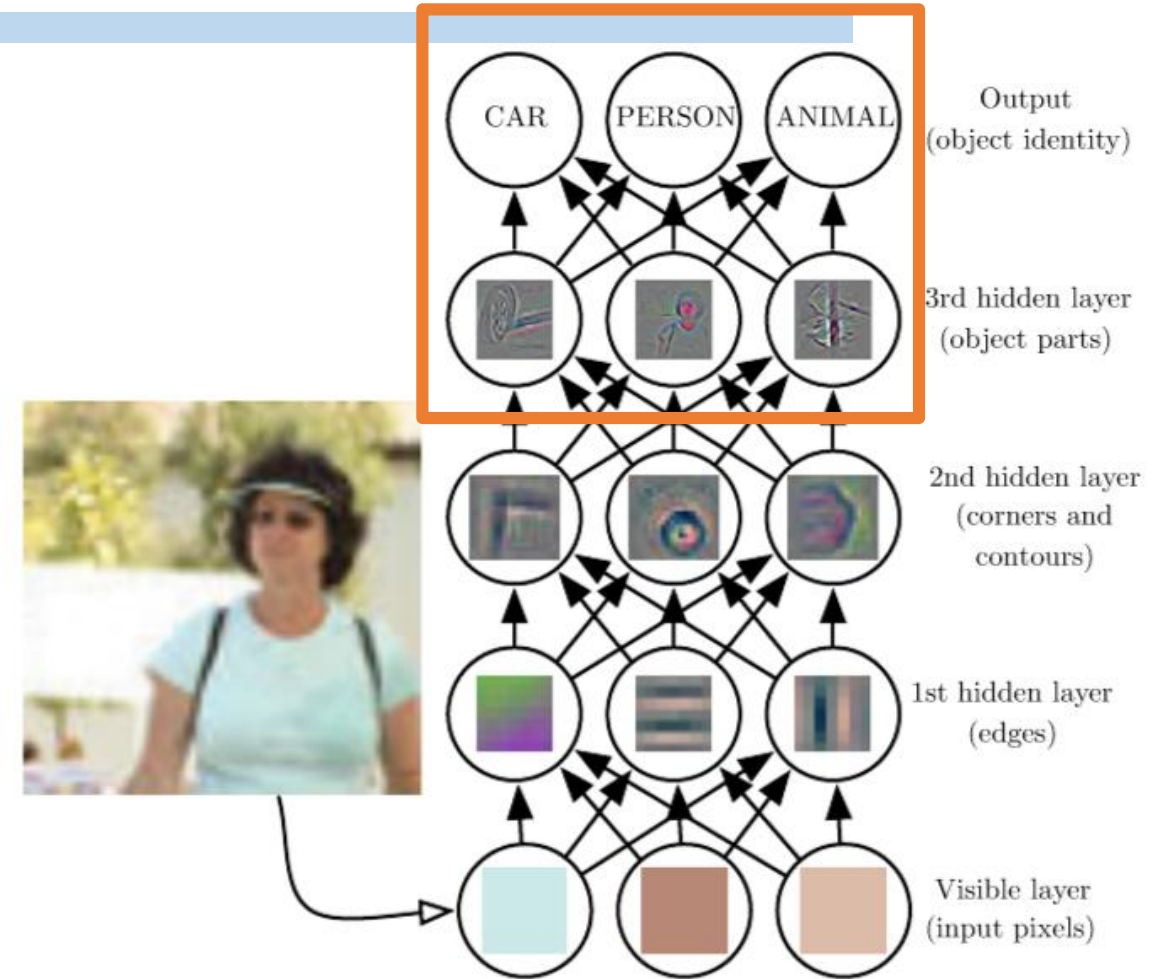
(<https://adeshpande3.github.io>)



(Deep Learning, MIT Press, 2016)

Brief Introduction to Deep Learning

Obtain *mapping* from representation to output.

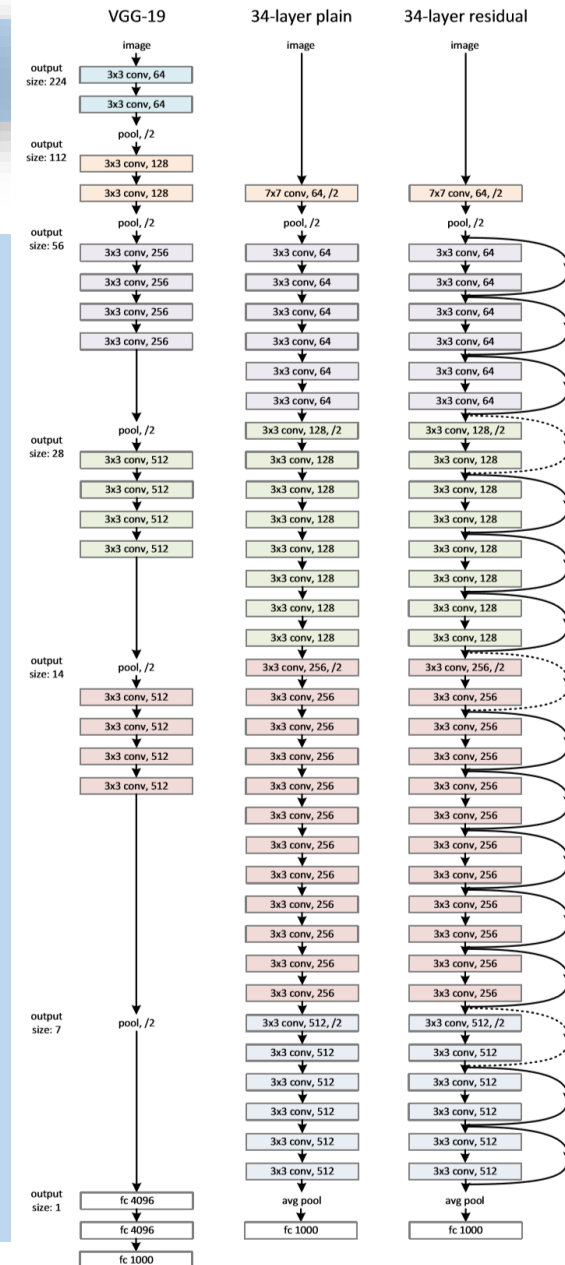


(Deep Learning, MIT Press, 2016)

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

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

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

Brief Introduction to Deep Learning

What can deep learning do in Earth science?

Classification

Denoising

Forward Modeling

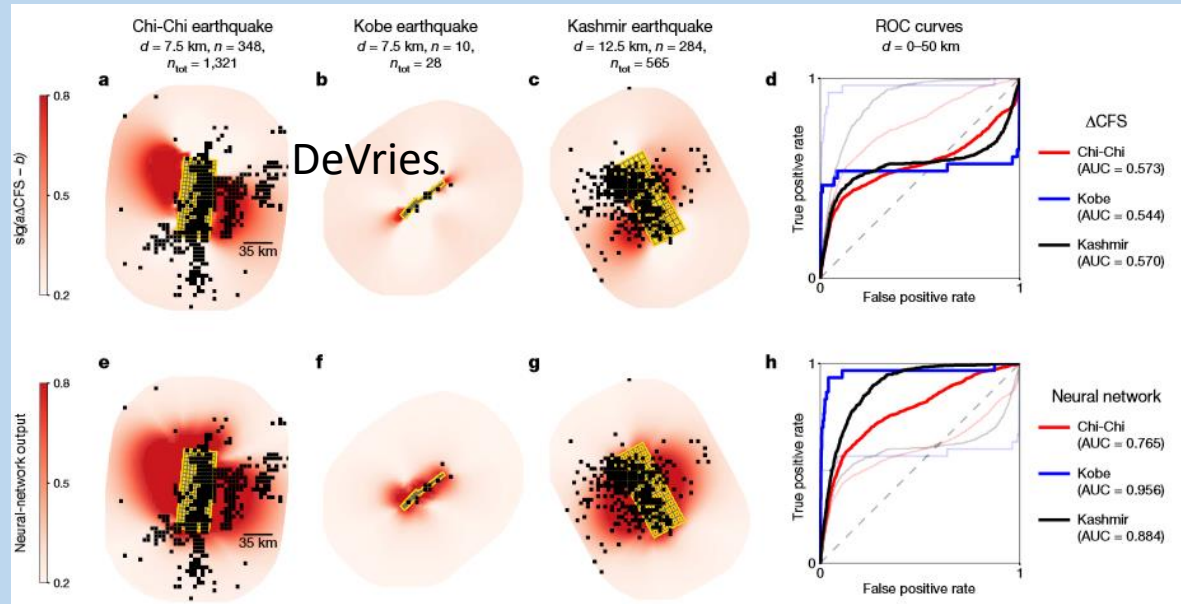
Inversion

...

What can deep learning do in Earth science?

Classification

Predicting aftershocks
following large earthquakes



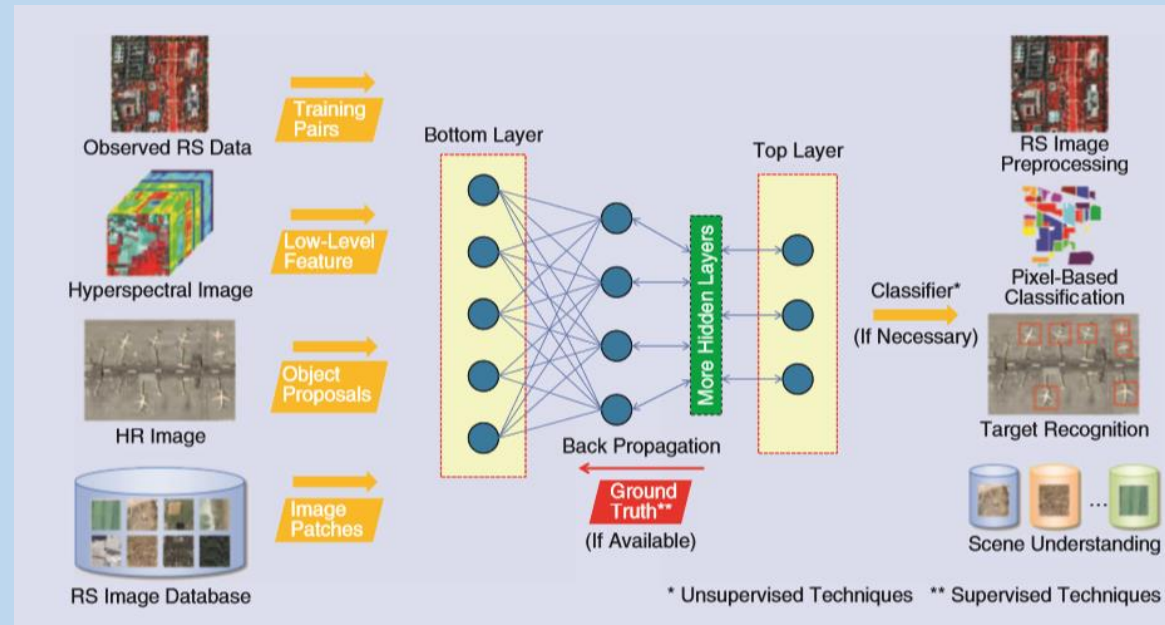
(DeVries et al., 2018)

Brief Introduction to Deep Learning

What can deep learning do in Earth science?

Classification

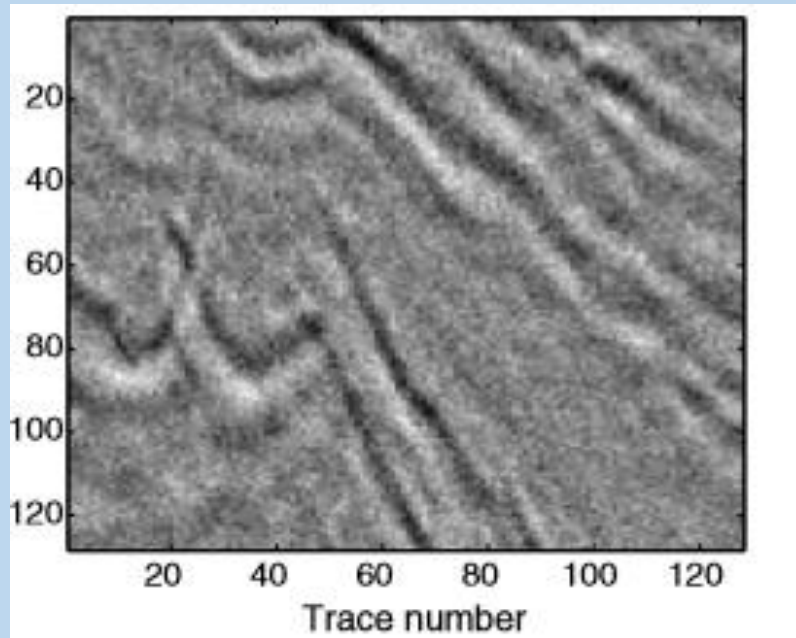
Processing remote sensing data



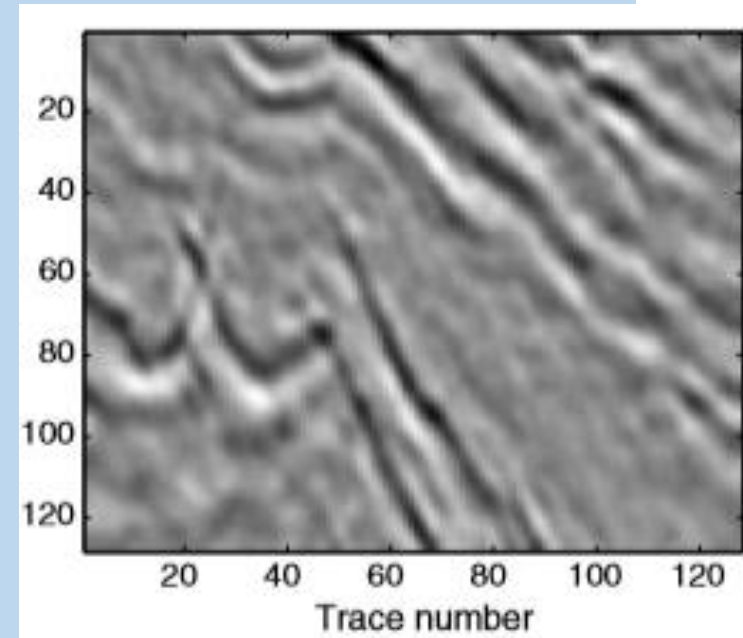
(Zhang et al., 2016)

What can deep learning do in Earth science?

Denoising



Noisy input



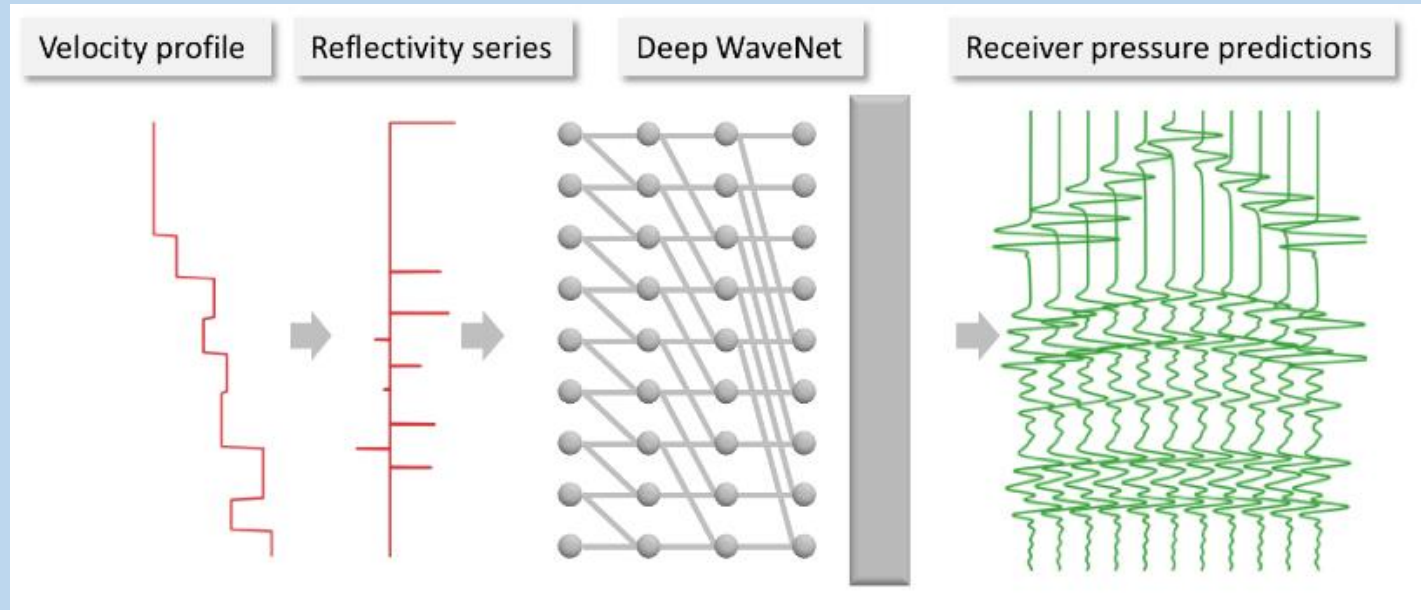
DL output

(Beckouche and Ma, 2014)

What can deep learning do in Earth science?

Forward Modeling

Fast approximate
simulation of
seismic waves with
deep learning

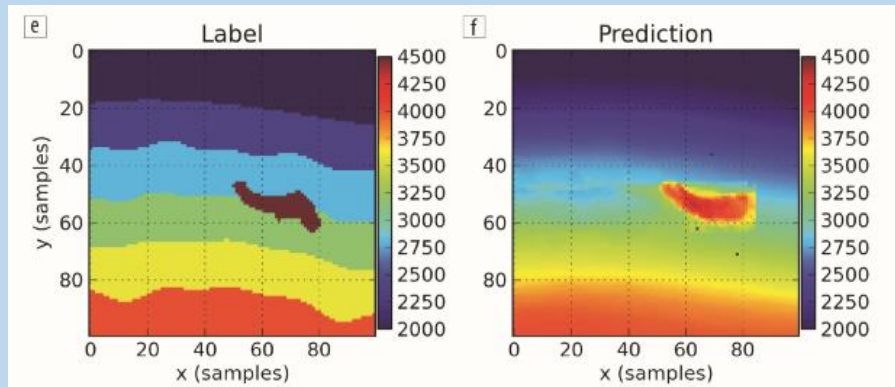


(Moseley et al., 2018)

Brief Introduction to Deep Learning

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

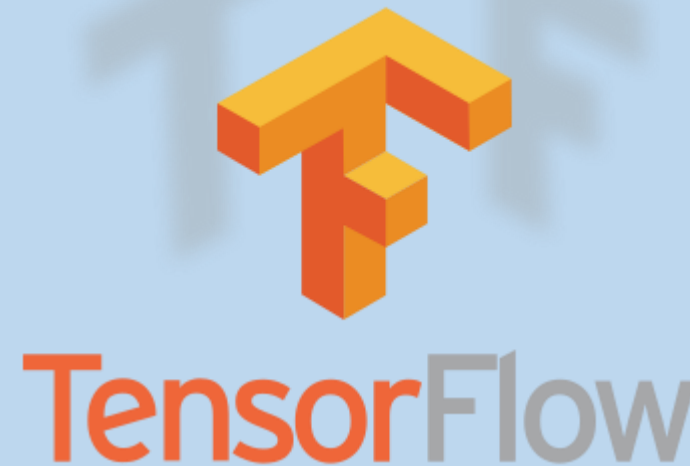
TensorFlow Basics

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/>)

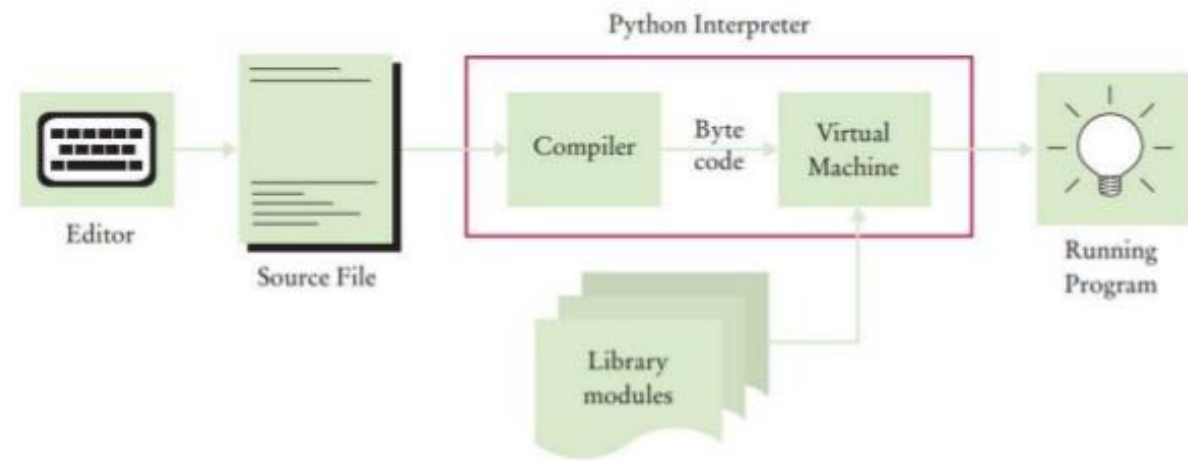
TensorFlow Basics

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



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

How The Python Interpreter Works



(<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/>



Image Manipulation with skimage

This example builds a simple UI for performing basic image manipulation with [scikit-image](#).

```
In [21]: from ipywidgets import interact, interactive, fixed
         from IPython.display import display
```

```
In [22]: import skimage
         from skimage import data, filter, io
```

```
In [23]: i = data.coffee()
```

```
In [24]: io.Image(i)
```

Out[24]:



```
In [25]: def edit_image(image, sigma=0.1, r=1.0, g=1.0, b=1.0):
         new_image = filter.gaussian_filter(image, sigma=sigma, multichannel=True)
         new_image[:, :, 0] = r * new_image[:, :, 0]
         new_image[:, :, 1] = g * new_image[:, :, 1]
         new_image[:, :, 2] = b * new_image[:, :, 2]
         new_image = io.Image(new_image)
         display(new_image)
         return new_image
```

```
In [26]: lims = (0.0, 1.0, 0.01)
         w = interactive(edit_image, image=fixed(i), sigma=(0.0, 10.0, 0.1), r=lims, g=lims, b=lims)
         display(w)
```



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 Basics

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)

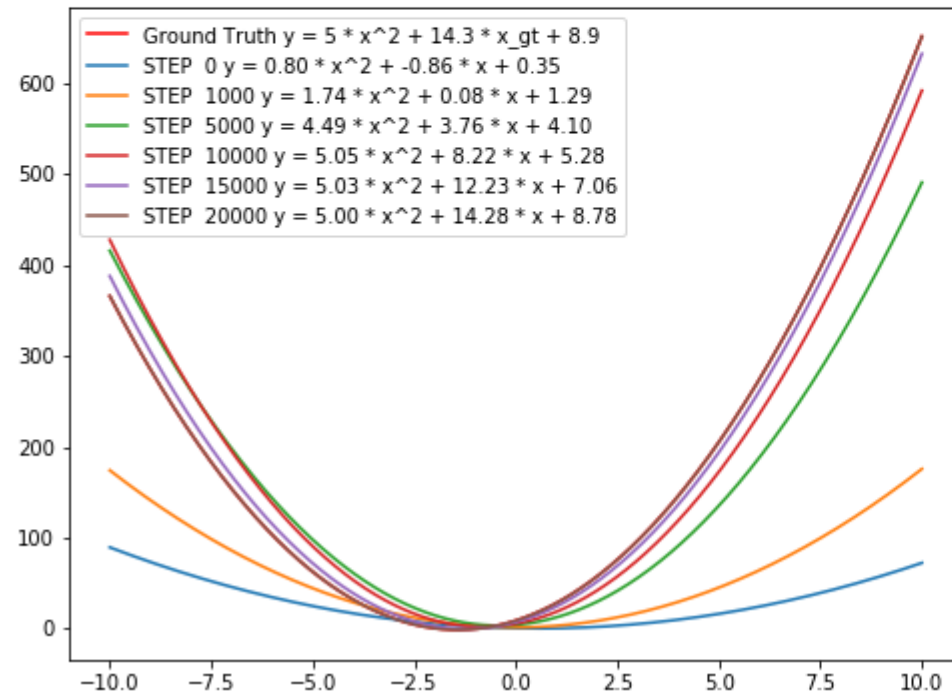
[[-1.8857789  0.02845232  2.23009   ]
 [ 0.20160252  0.49441913  0.37605742]
 [ 3.5984905  1.7590961 -0.84973013]]
```

TensorFlow Basics

Approximate Quadratic Function With TensorFlow

```
: 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

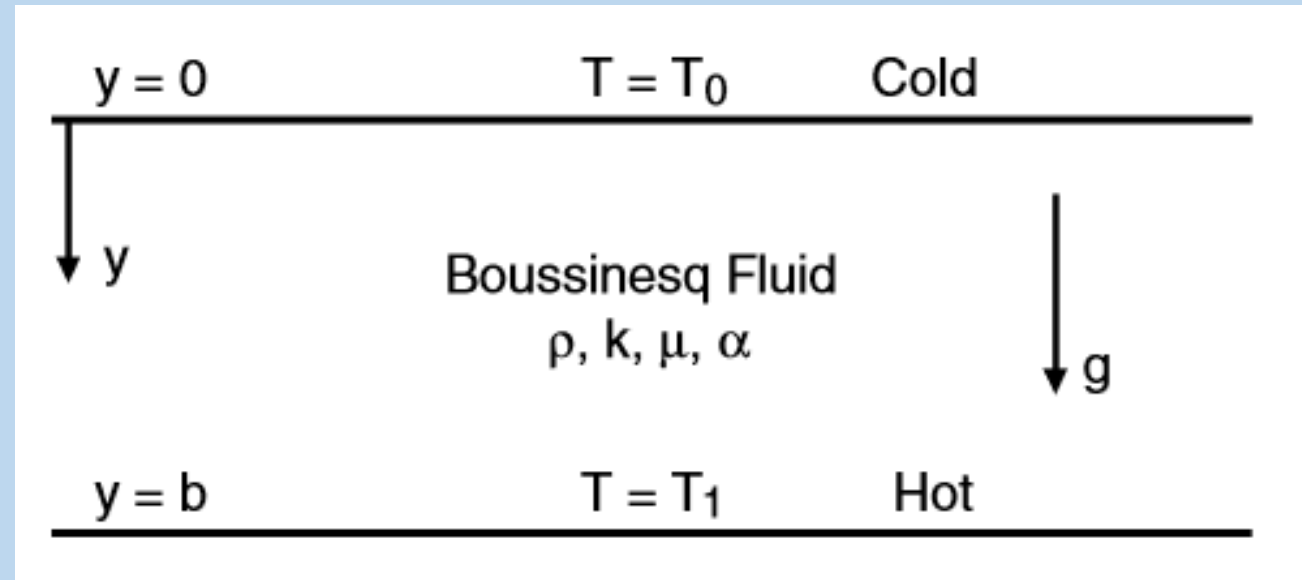
3

**Classifying Stability of Mantle with
Neural Networks: An Example**

4

Discussions

Plane Layer Heated from Below



(Schuber et al., 2001)



Classifying Stability of Mantle with Neural Networks: An Example



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]  $\lambda$ 
    #instance[7]  $(T_0 - T_1)/1000$ 
    #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  $\lambda$ 
    data_set[:, 6] = np.random.uniform(0.0, 6.0, size=instance_num)
    #simulate  $T_0 - T_1$ 
    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[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_training_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 training
for idx in range(step_num):
    input_batch, label_batch = get_training_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
0.34709388
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']})))
```

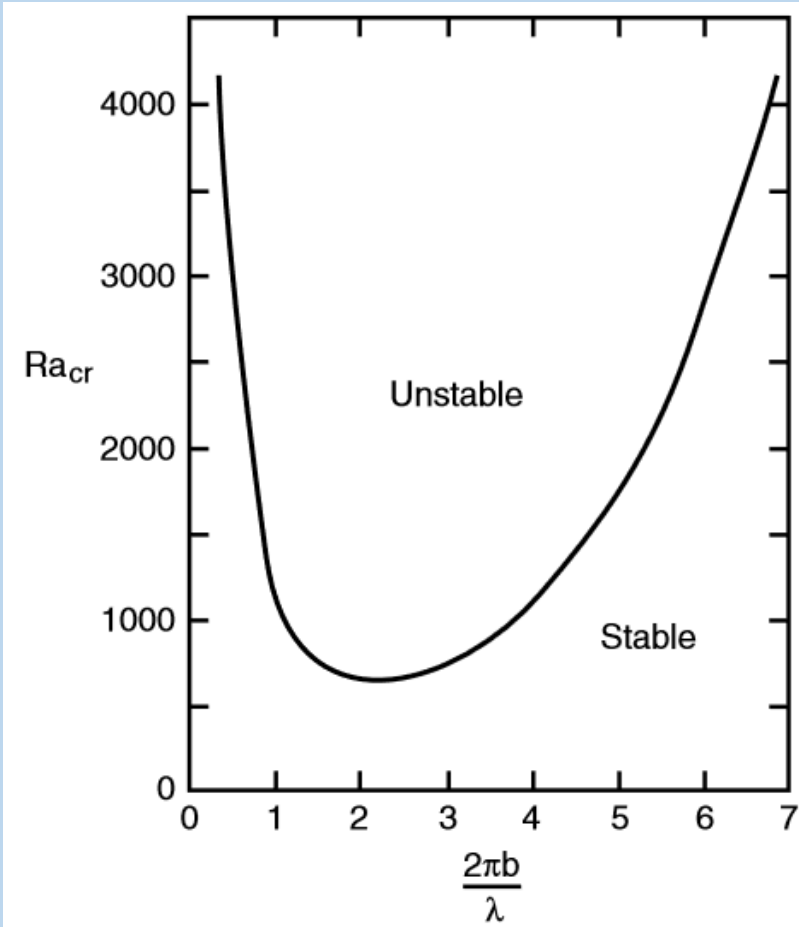
```
Accuracy: 0.9649959802627563
```

Classifying Stability of Mantle with Neural Networks: An Example

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

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

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



(Schuber et al., 2001)



Classifying Stability of Mantle with Neural Networks: An Example



OUTLINES

1

Brief Introduction to Deep Learning

2

TensorFlow Basics

3

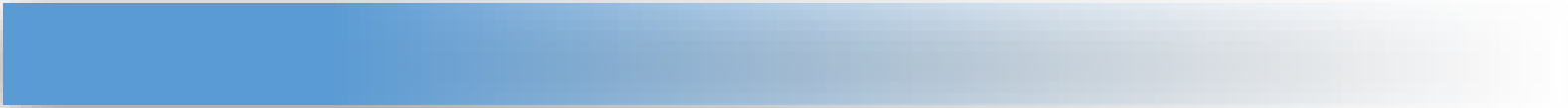
**Classifying Stability of Mantle with
Neural Networks: An Example**

4

Discussions

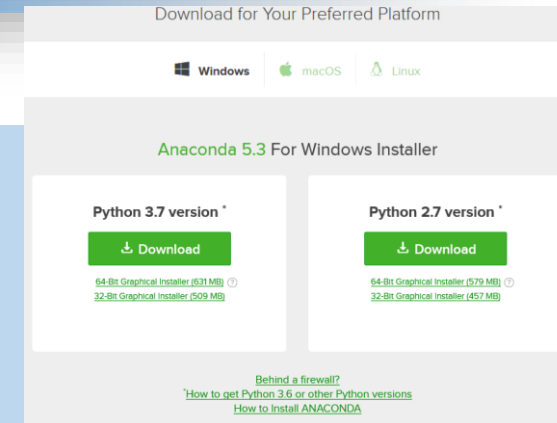


Discussions



TensorFlow Installation via Anaconda

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



Step 2. Create a new conda environment containing TensorFlow.

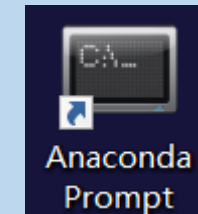
Open Anaconda Prompt and run

'conda create -n tensorflow_env tensorflow python=3.6'

or

'conda create -n tensorflow_gpuenv tensorflow-gpu python=3.6'

for GPU version



Congratulations...

```
werkzeug-0.14.1 100% ##### Time: 0:00:00 2.93 MB/
wincertstore-0 100% ##### Time: 0:00:00 1.64 MB/
absl-py-0.6.1- 100% ##### Time: 0:00:00 3.16 MB/
setuptools-40. 100% ##### Time: 0:00:00 3.09 MB/
grpcio-1.14.1- 100% ##### Time: 0:00:00 2.79 MB/
protobuf-3.6.1 100% ##### Time: 0:00:00 3.13 MB/
wheel-0.32.2-p 100% ##### Time: 0:00:00 2.03 MB/
pip-18.1-py36_ 100% ##### Time: 0:00:02 846.26 KB/
mkl_fft-1.0.6- 100% ##### Time: 0:00:00 3.18 MB/
mkl_random-1.0 100% ##### Time: 0:00:00 3.22 MB/
numpy-1.15.4-p 100% ##### Time: 0:00:00 921.88 kB/
tensorboard-1. 100% ##### Time: 0:00:01 3.04 MB/
tensorflow-1.1 100% ##### Time: 0:00:11 3.03 MB/
#
```

```
# To activate this environment, use:
# > activate tensorflow_env
#
# To deactivate an active environment, use:
# > deactivate
#
# * for power-users using bash, you must source
```

无标题



References

