TAIPEI TECH

Lenet實作解說

Speaker:Jerry

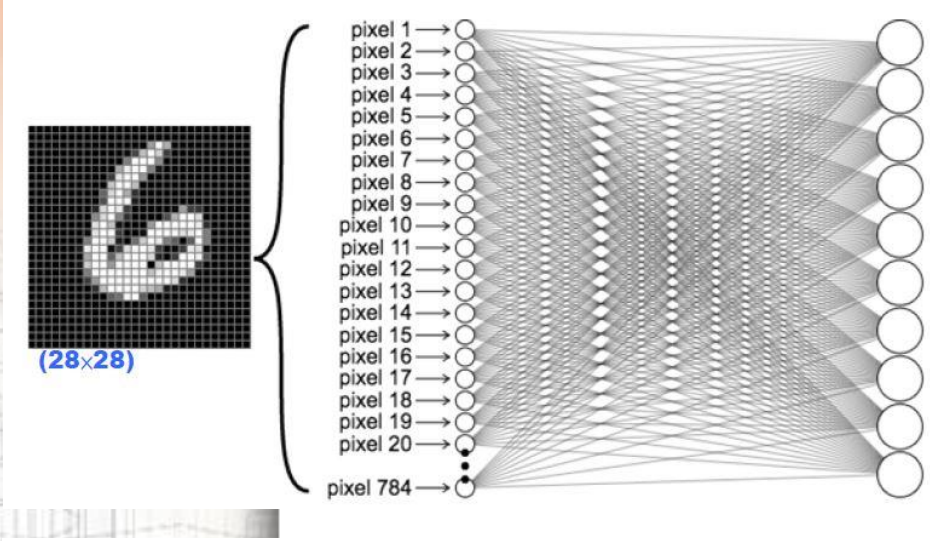
Date:2019/6/22

目錄

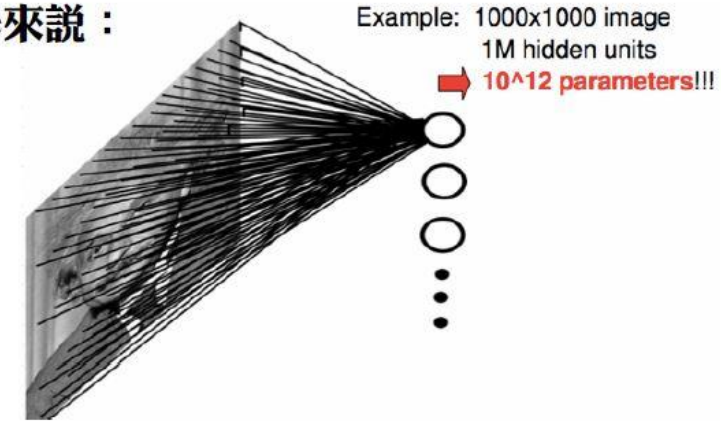
TAIPEI TECH

簡介

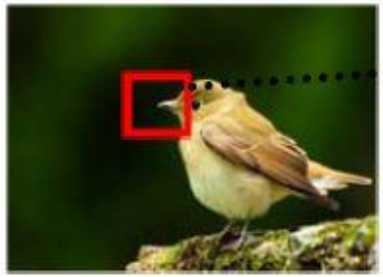
Fully Connected NN



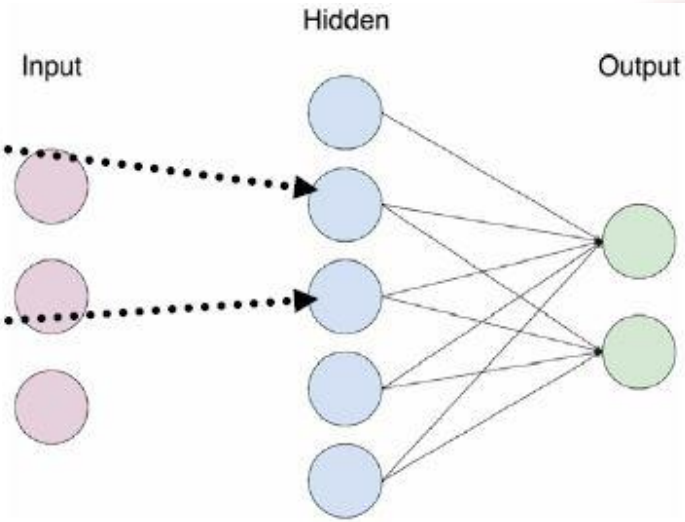
以一張具百萬像素的image來說：

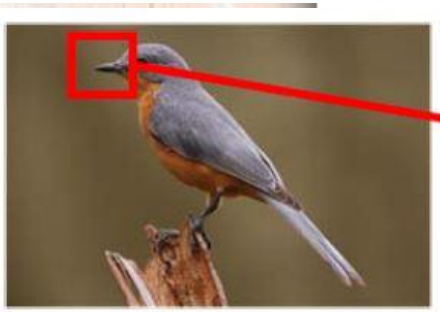


上方鳥嘴偵測



中間鳥嘴偵測

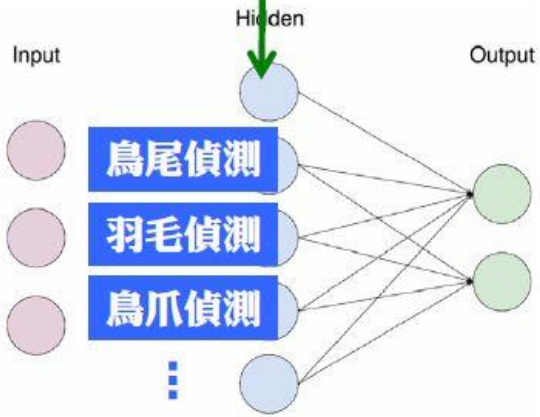




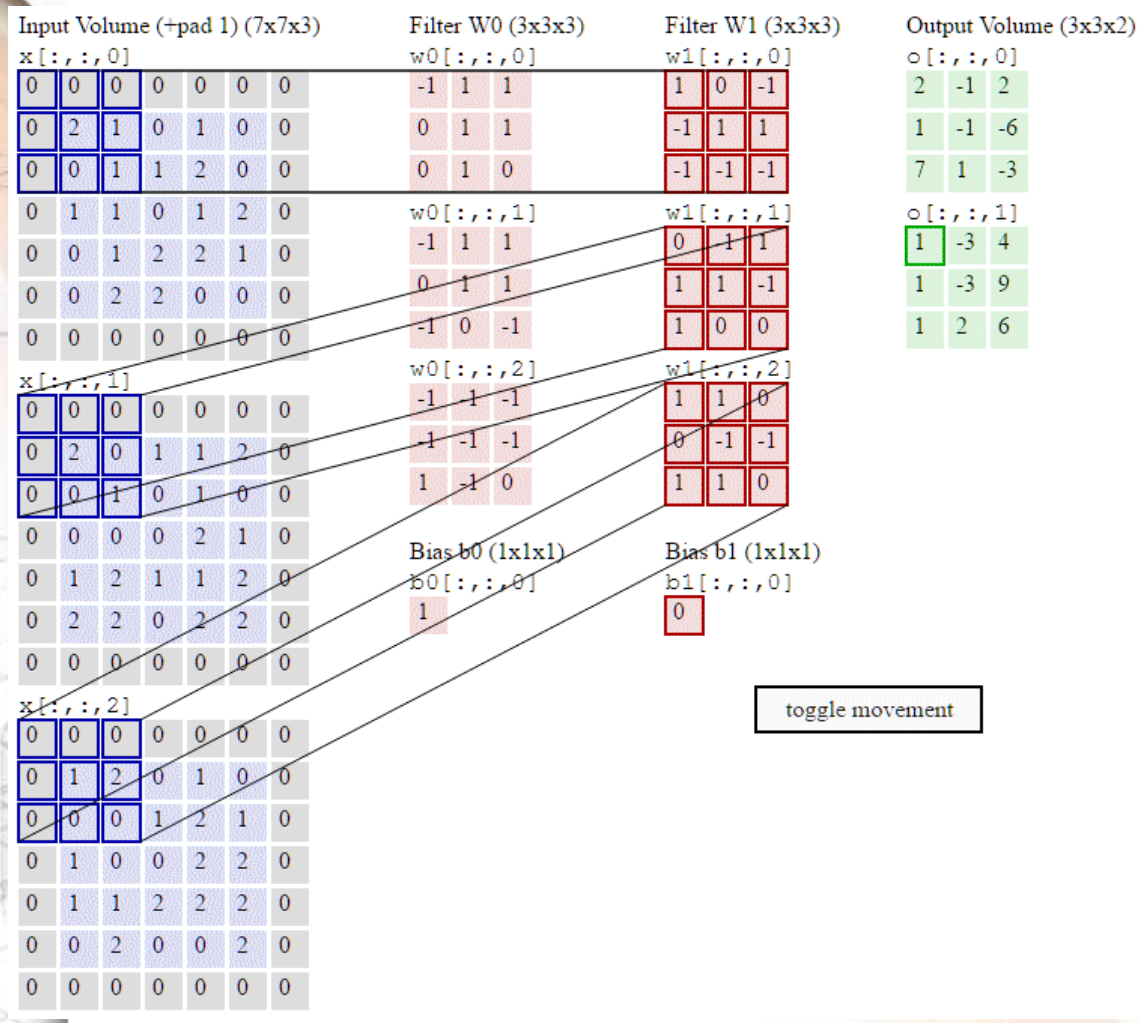
鳥嘴偵測

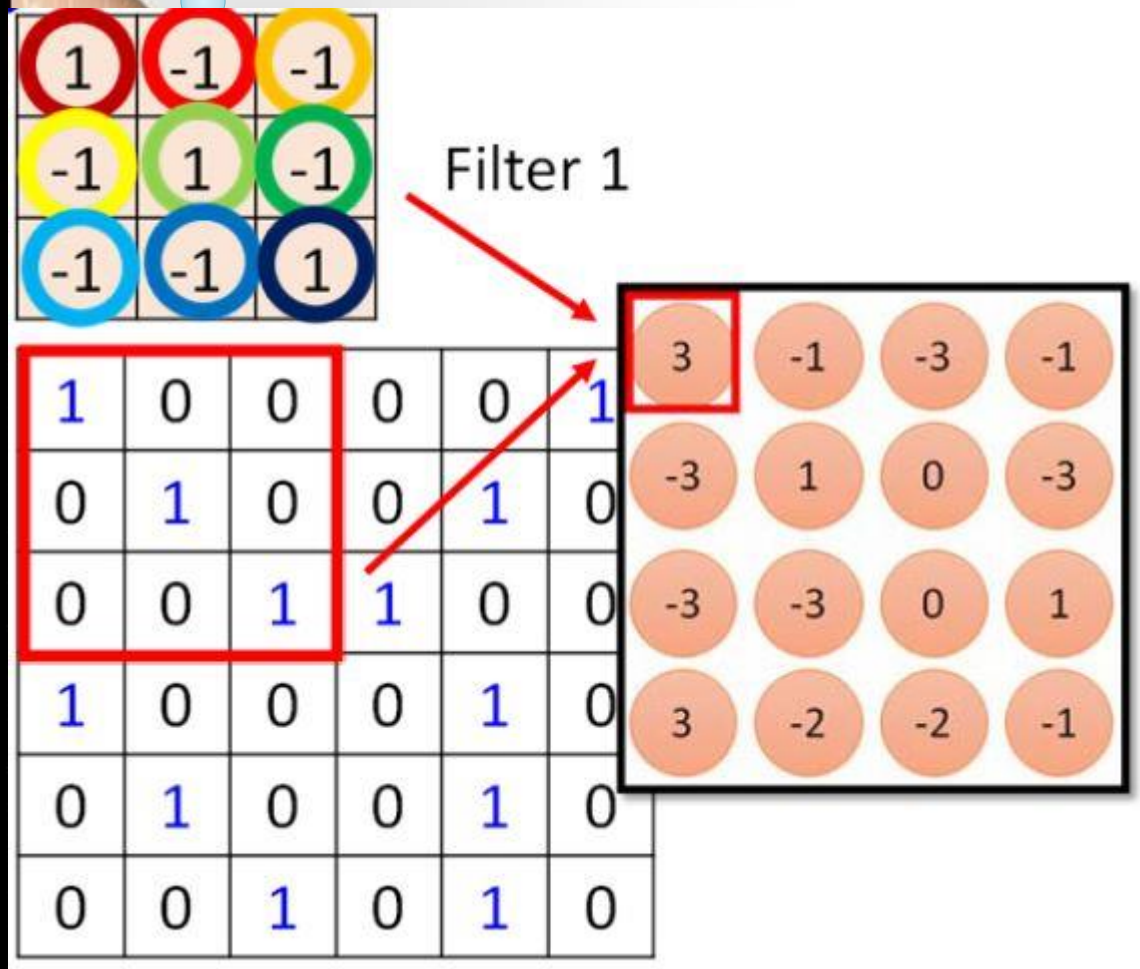
每個節點僅需以較少的參數，連結部份區域。

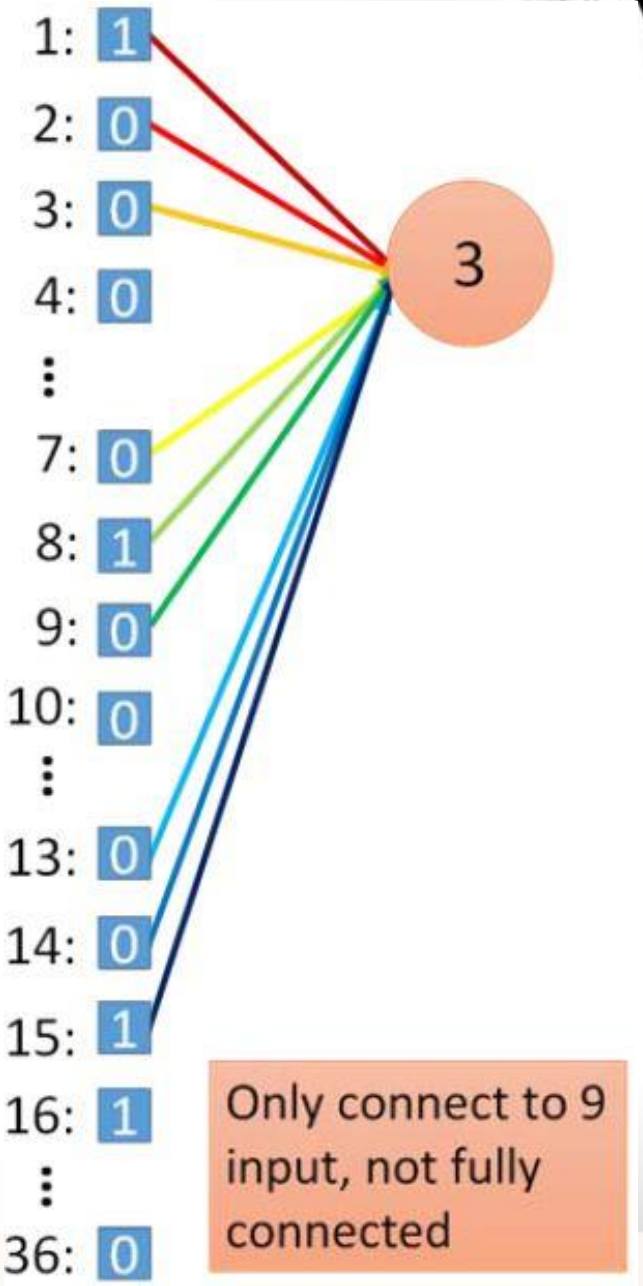
所有節點的偵測結果合併起來後，就可以得知圖片中是否有一隻鳥。



Receptive Field and Shared weight



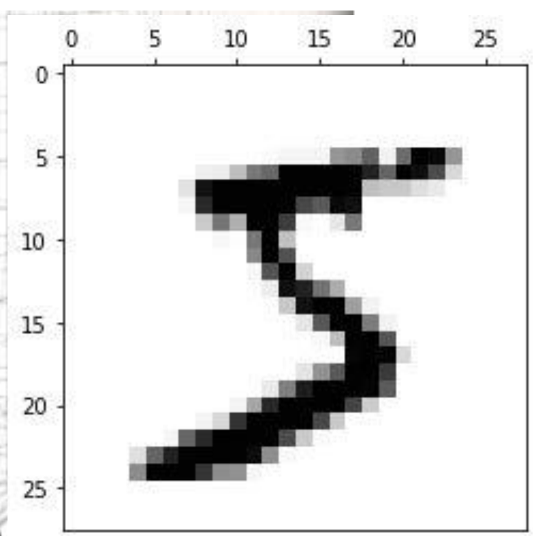

1 -1 -1
-1 1 -1
-1 -1 1
Filter 1
1 0 0 0 0 1
0 1 0 0 1 0
0 0 1 1 0 0
1 0 0 0 1 0
0 1 0 0 1 0
0 0 1 0 1 0
3 -1 -3 -1
-3 1 0 -3
-3 -3 0 1
3 -2 -2 -1

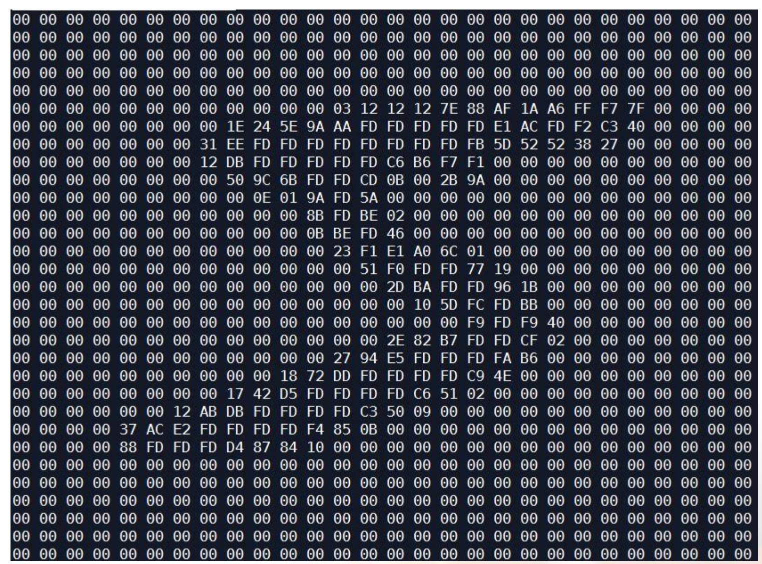

1: 1
2: 0
3: 0
4: 0
7: 0
8: 1
9: 0
10: 0
13: 0
14: 0
15: 1
16: 1
36: 0
3
Only connect to 9 input, not fully connected

MNIST Dataset

Four files are available on this site:

train-images-idx3-ubyte.gz: training set images (9912422 bytes)
train-labels-idx1-ubyte.gz: training set labels (28881 bytes)
t10k-images-idx3-ubyte.gz: test set images (1648877 bytes)
t10k-labels-idx1-ubyte.gz: test set labels (4542 bytes)

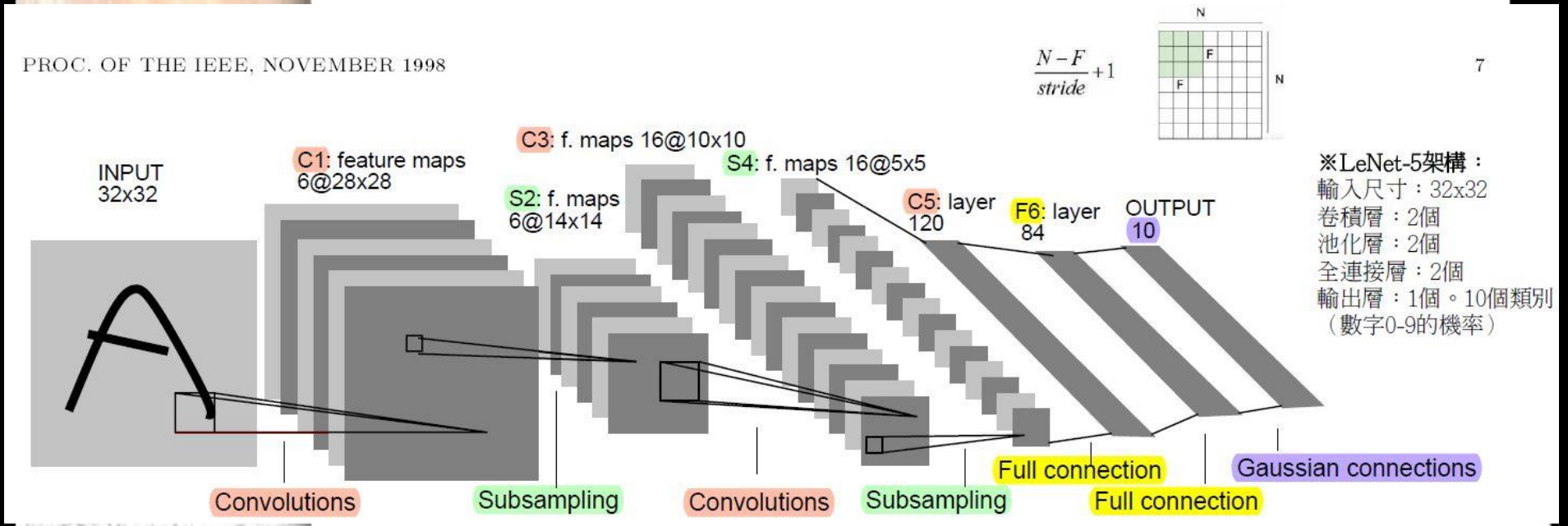




```
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 03 12 12 12 7E 88 AF 1A A6 FF F7 7F 00 00 00 00
00 00 00 00 00 00 00 00 1E 24 5E 9A AA FD FD FD FD FD E1 AC FD F2 C3 40 00 00 00 00
00 00 00 00 00 00 00 31 EE FD FD FD FD FD FD FD FD FB 5D 52 52 38 27 00 00 00 00 00
00 00 00 00 00 00 00 12 DB FD FD FD FD FD C6 B6 F7 F1 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 50 9C 6B FD FD CD 0B 00 2B 9A 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 0E 01 9A FD 5A 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 8B FD BE 02 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 0B BE FD 46 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 23 F1 E1 A0 6C 01 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 51 F0 FD FD 77 19 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 2D BA FD FD 96 1B 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 10 5D FC FD BB 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 F9 FD F9 40 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 2E 82 B7 FD FD CF 02 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 27 94 E5 FD FD FD FA B6 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 18 72 DD FD FD FD FD C9 4E 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 17 42 D5 FD FD FD FD C6 51 02 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 12 AB DB FD FD FD FD C3 50 09 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 37 AC E2 FD FD FD FD F4 85 0B 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 88 FD FD FD D4 87 84 10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```


http://yann.lecun.com/exdb/mnist/

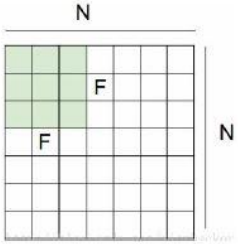
Lenet架構

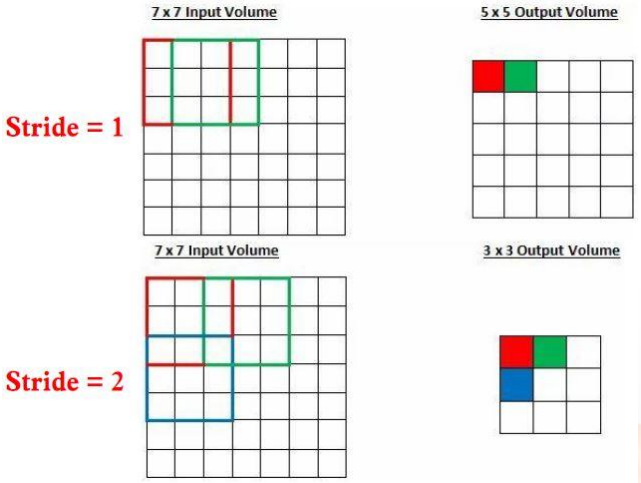

PROC. OF THE IEEE, NOVEMBER 1998
7
$\frac{N-F}{stride}+1$
N
F
F
N
INPUT
32x32
C1: feature maps
6@28x28
S2: f. maps
6@14x14
C3: f. maps 16@10x10
S4: f. maps 16@5x5
C5: layer
120
F6: layer
84
OUTPUT
10
※LeNet-5架構：
輸入尺寸：32x32
卷積層：2個
池化層：2個
全連接層：2個
輸出層：1個。10個類別
（數字0-9的機率）
Convolutions
Subsampling
Convolutions
Subsampling
Full connection
Full connection
Gaussian connections

Layer		Feature Map	Size	Kernel Size	Stride	Activation
Input	Image	1	32x32	-	-	-
1	Convolution	6	28x28	5x5	1	tanh
2	Average Pooling	6	14x14	2x2	2	tanh
3	Convolution	16	10x10	5x5	1	tanh
4	Average Pooling	16	5x5	2x2	2	tanh
5	Convolution	120	1x1	5x5	1	tanh
6	FC	-	84	-	-	tanh
Output	FC	-	10	-	-	softmax

Output size at different stride…

$$\frac{N-F}{stride}+1$$



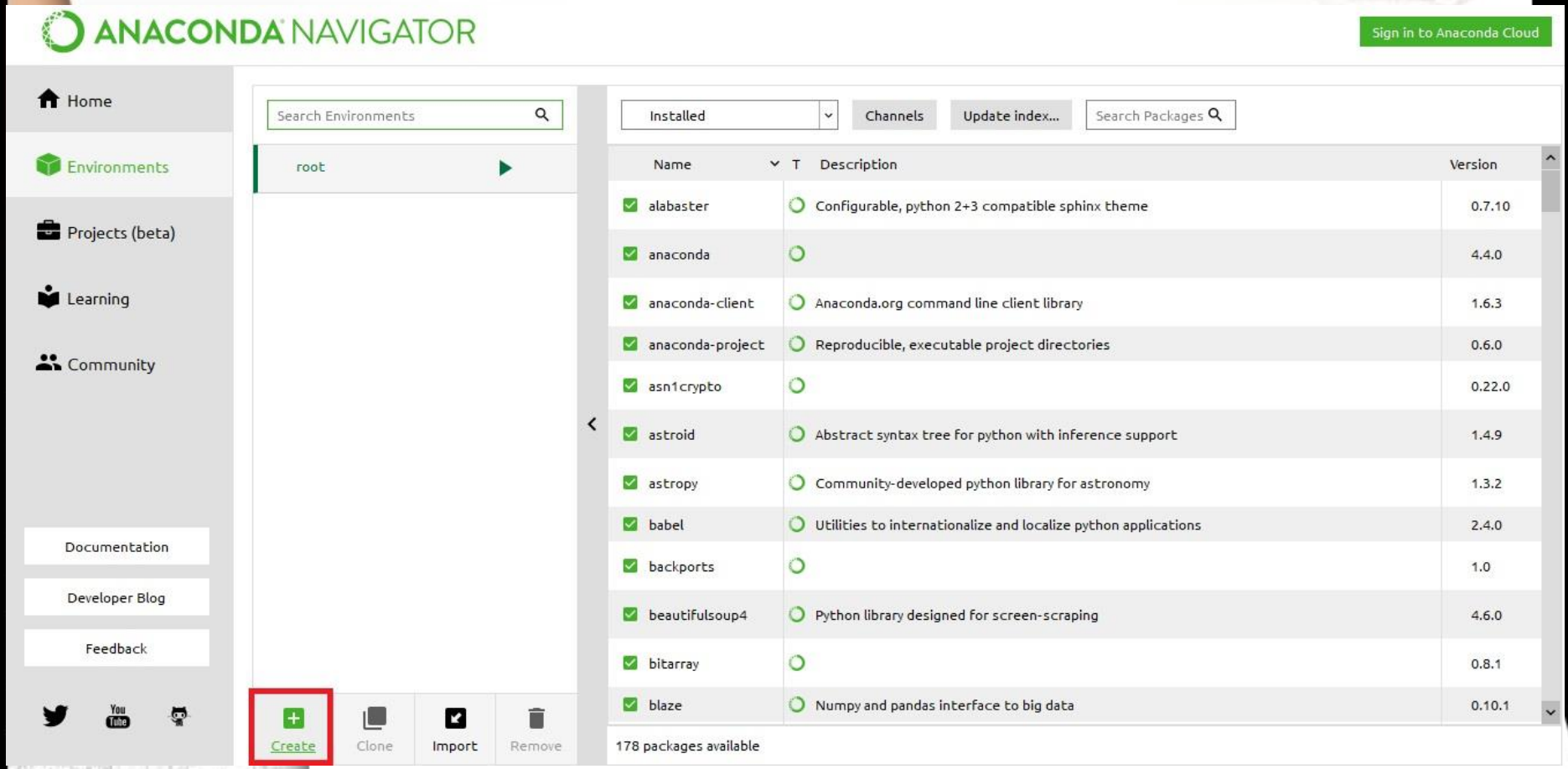


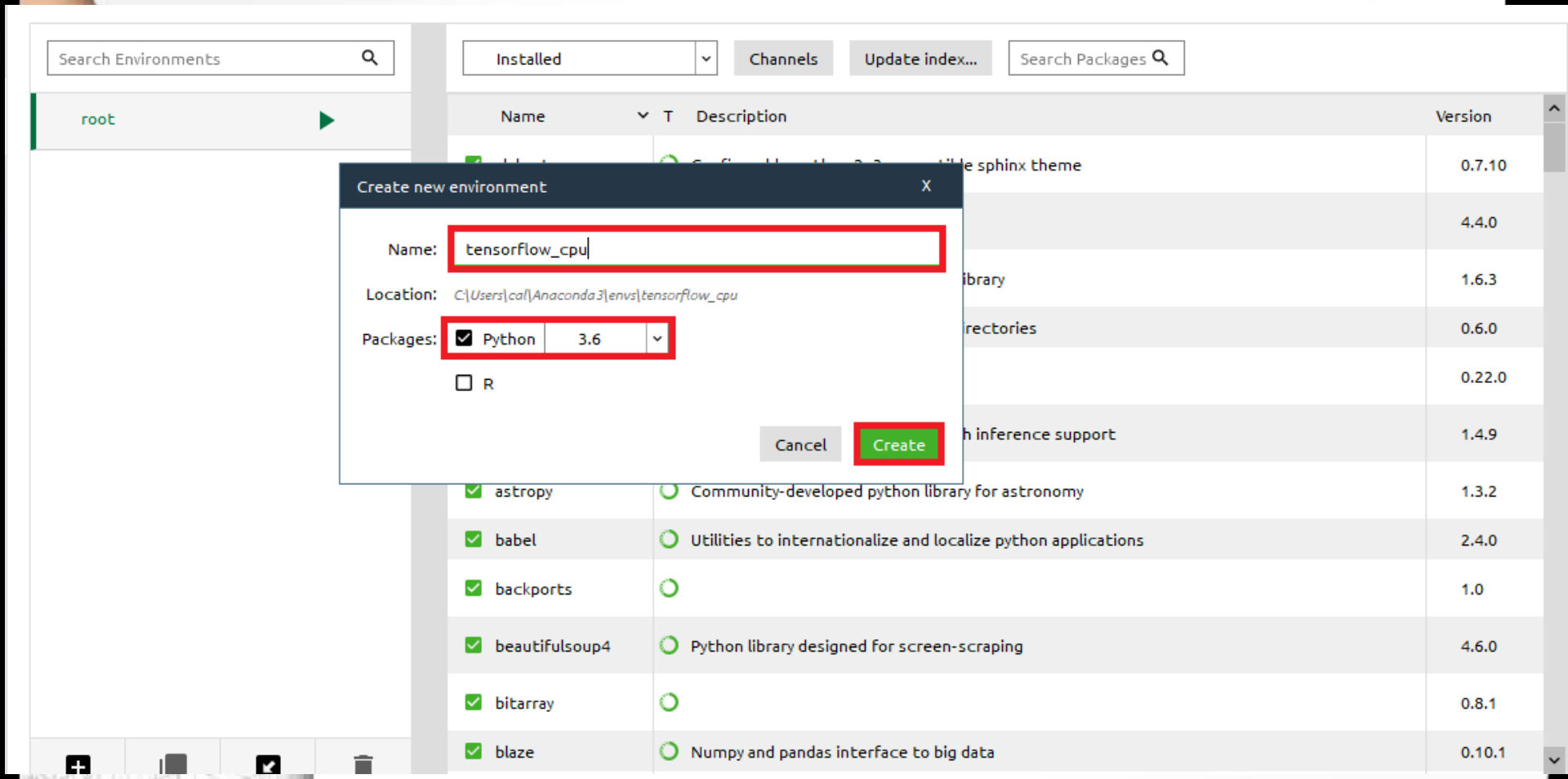
TAIPEI TECH

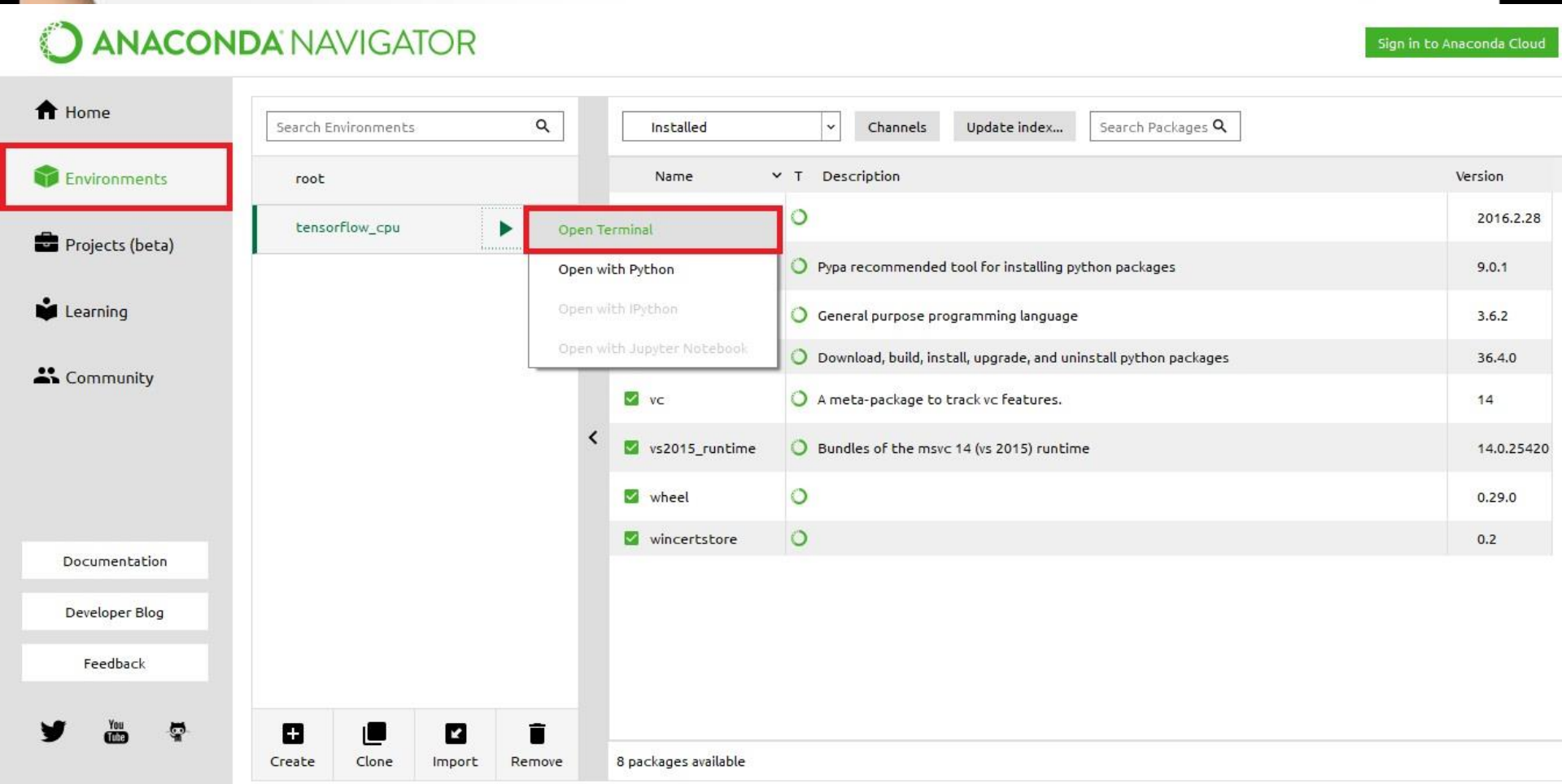
程式碼解說

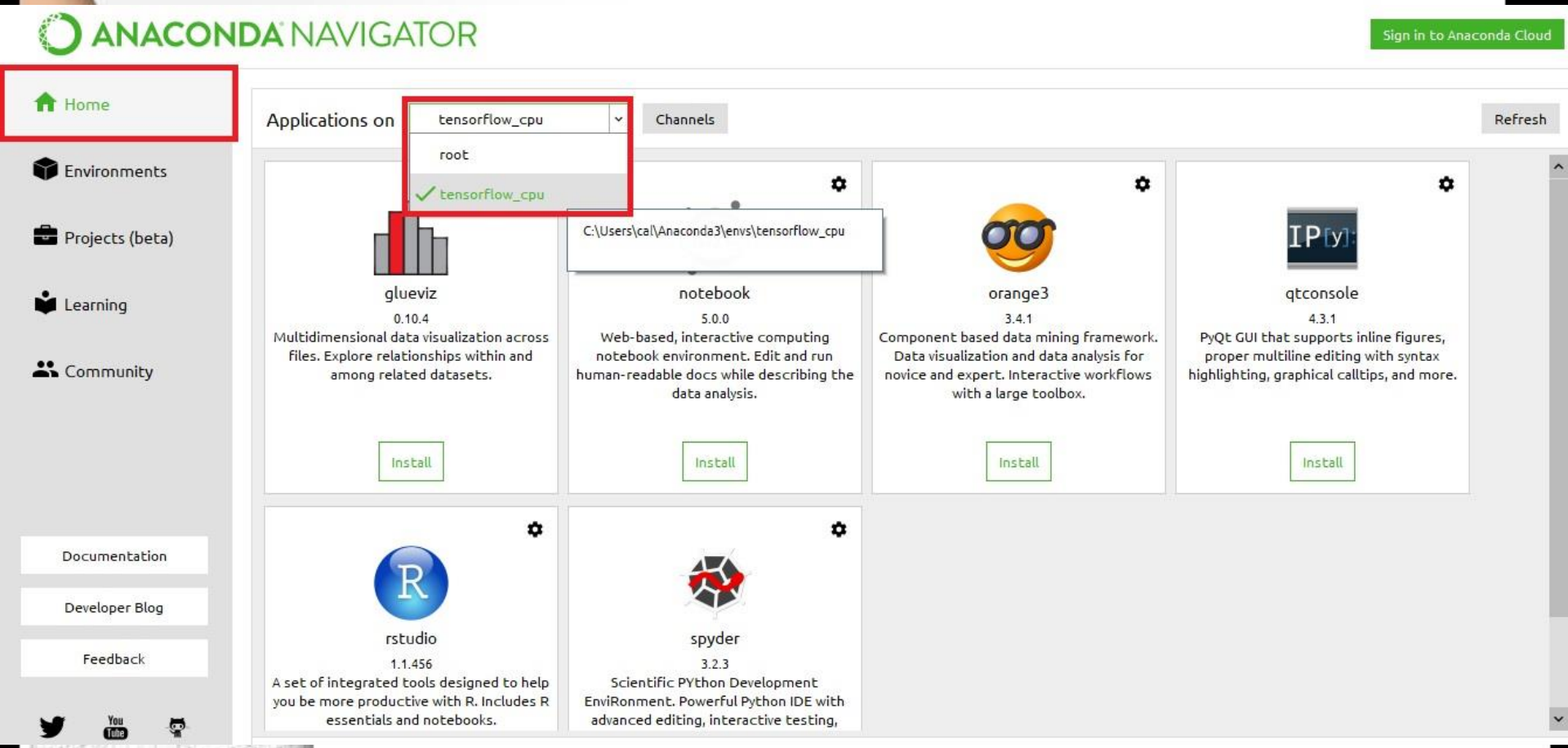
環境安裝

到官方網站下載Anaconda，安裝後可進入到下面步驟!!

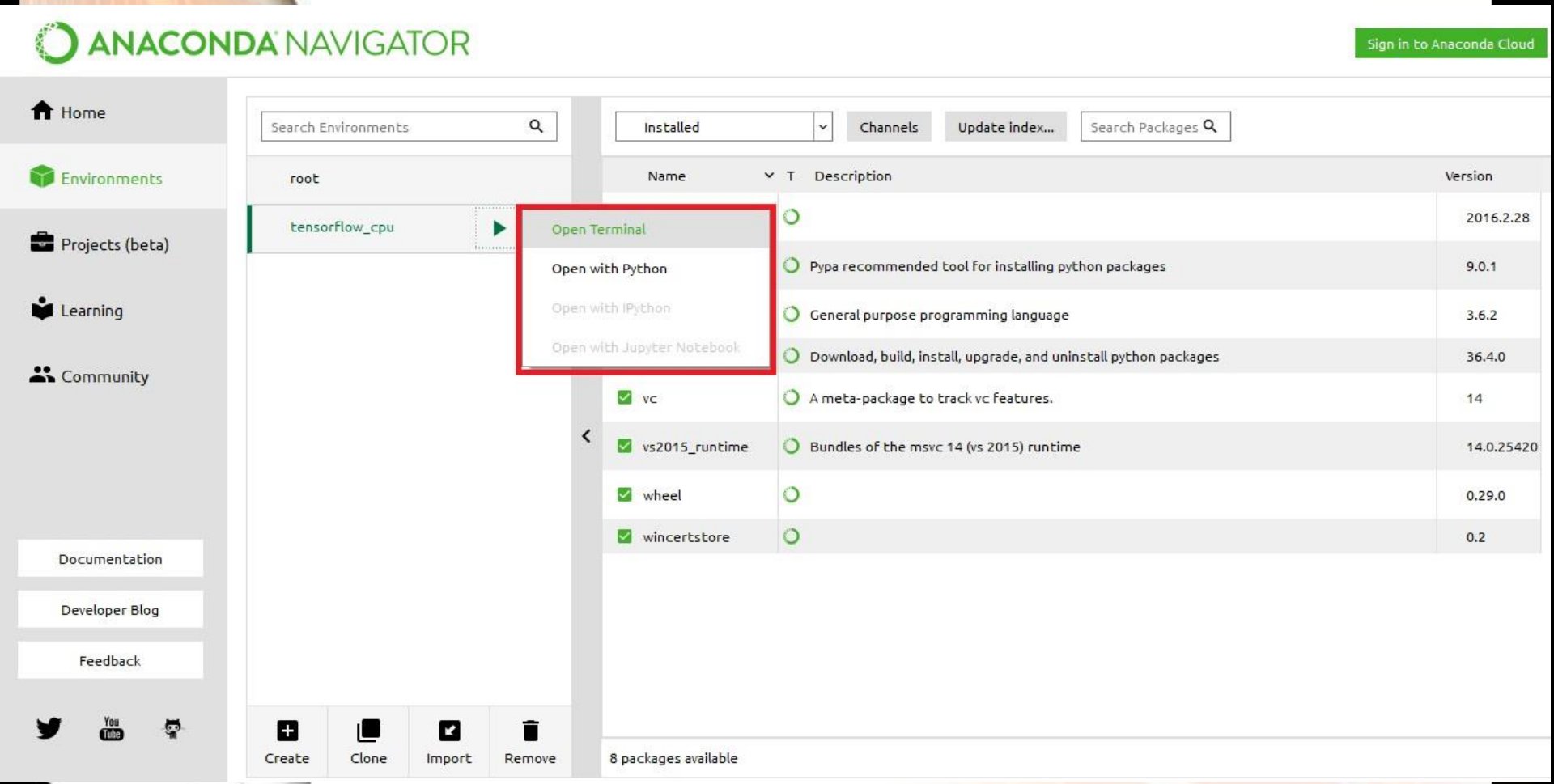

ANACONDA NAVIGATOR
Sign in to Anaconda Cloud
Home
Environments
Projects (beta)
Learning
Community
Documentation
Developer Blog
Feedback
Search Environments
root
Create
Clone
Import
Remove
Installed
Channels
Update index...
Search Packages
Name
T
Description
Version
alabaster
Configurable, python 2+3 compatible sphinx theme
0.7.10
anaconda
4.4.0
anaconda-client
Anaconda.org command line client library
1.6.3
anaconda-project
Reproducible, executable project directories
0.6.0
asn1crypto
0.22.0
astroid
Abstract syntax tree for python with inference support
1.4.9
astropy
Community-developed python library for astronomy
1.3.2
babel
Utilities to internationalize and localize python applications
2.4.0
backports
1.0
beautifulsoup4
Python library designed for screen-scraping
4.6.0
bitarray
0.8.1
blaze
Numpy and pandas interface to big data
0.10.1
178 packages available


Search Environments
root
Installed
Channels
Update index...
Search Packages
Name
T
Description
Version
Create new environment
X
Name:
tensorflow_cpu
Location:
C:\Users\cal\Anaconda3\envs\tensorflow_cpu
Packages:
Python
3.6
R
Cancel
Create
sphinx theme
0.7.10
4.4.0
ibrary
1.6.3
irectories
0.6.0
0.22.0
h inference support
1.4.9
astropy
Community-developed python library for astronomy
1.3.2
babel
Utilities to internationalize and localize python applications
2.4.0
backports
1.0
beautifulsoup4
Python library designed for screen-scraping
4.6.0
bitarray
0.8.1
blaze
Numpy and pandas interface to big data
0.10.1


ANACONDA NAVIGATOR
Sign in to Anaconda Cloud
Home
Environments
Projects (beta)
Learning
Community
Documentation
Developer Blog
Feedback
Search Environments
root
tensorflow_cpu
Open Terminal
Open with Python
Open with IPython
Open with Jupyter Notebook
Create
Clone
Import
Remove
Installed
Channels
Update index...
Search Packages
Name
T
Description
Version
2016.2.28
Pypa recommended tool for installing python packages
9.0.1
General purpose programming language
3.6.2
Download, build, install, upgrade, and uninstall python packages
36.4.0
vc
A meta-package to track vc features.
14
vs2015_runtime
Bundles of the msvc 14 (vs 2015) runtime
14.0.25420
wheel
0.29.0
wincertstore
0.2
8 packages available


ANACONDA NAVIGATOR
Sign in to Anaconda Cloud
Home
Environments
Projects (beta)
Learning
Community
Documentation
Developer Blog
Feedback
Applications on
tensorflow_cpu
Channels
Refresh
root
tensorflow_cpu
C:\Users\cal\Anaconda3\envs\tensorflow_cpu
glueviz
0.10.4
Multidimensional data visualization across files. Explore relationships within and among related datasets.
Install
notebook
5.0.0
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.
Install
orange3
3.4.1
Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.
Install
qtconsole
4.3.1
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.
Install
rstudio
1.1.456
A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.
spyder
3.2.3
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing,

打開Terninal

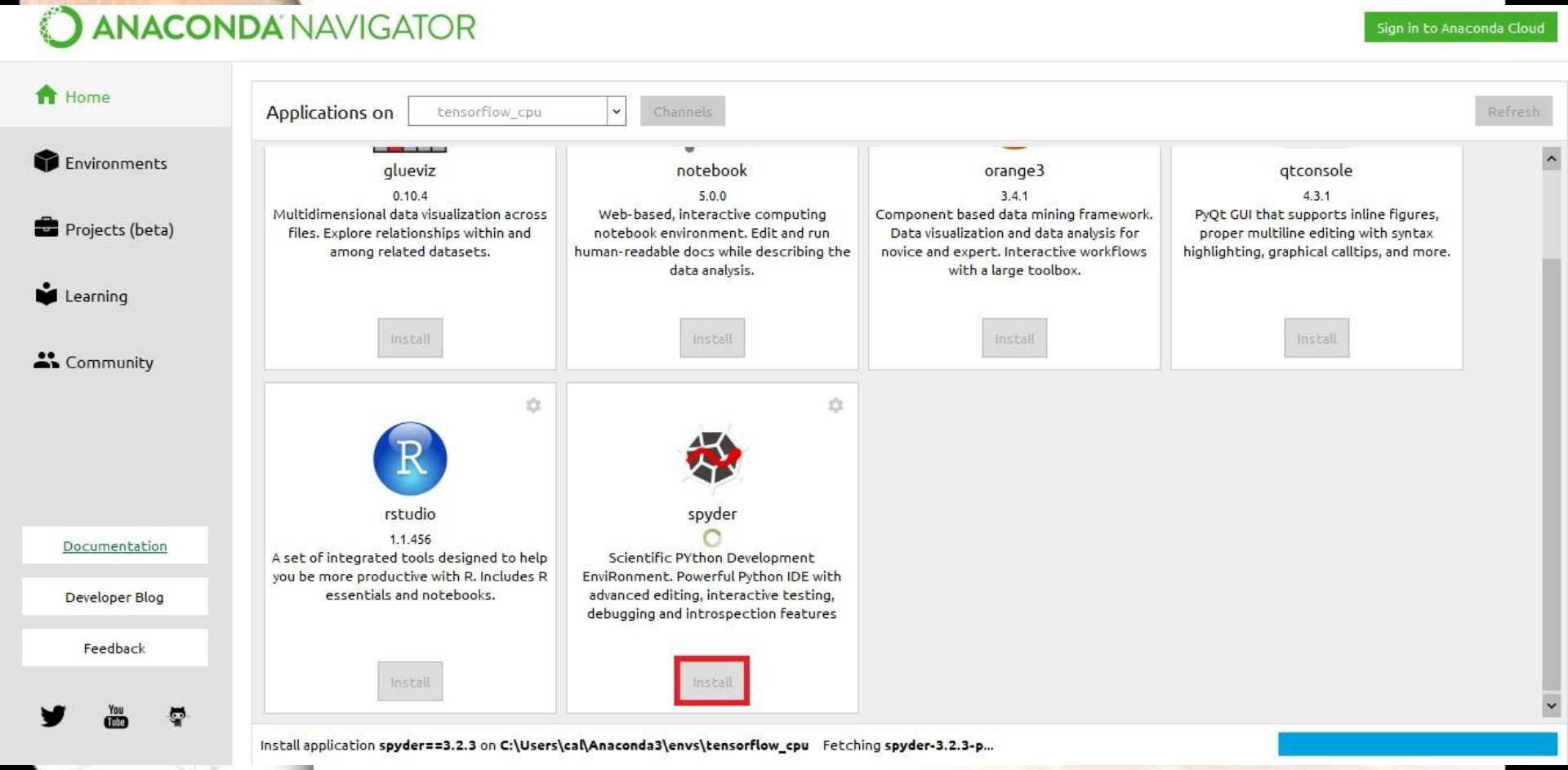

ANACONDA NAVIGATOR
Sign in to Anaconda Cloud
Home
Environments
Projects (beta)
Learning
Community
Documentation
Developer Blog
Feedback
Search Environments
root
tensorflow_cpu
Open Terminal
Open with Python
Open with IPython
Open with Jupyter Notebook
Installed
Channels
Update index...
Search Packages
Name
T
Description
Version
2016.2.28
Pypa recommended tool for installing python packages
9.0.1
General purpose programming language
3.6.2
Download, build, install, upgrade, and uninstall python packages
36.4.0
vc
A meta-package to track vc features.
14
vs2015_runtime
Bundles of the msvc 14 (vs 2015) runtime
14.0.25420
wheel
0.29.0
wincertstore
0.2
Create
Clone
Import
Remove
8 packages available

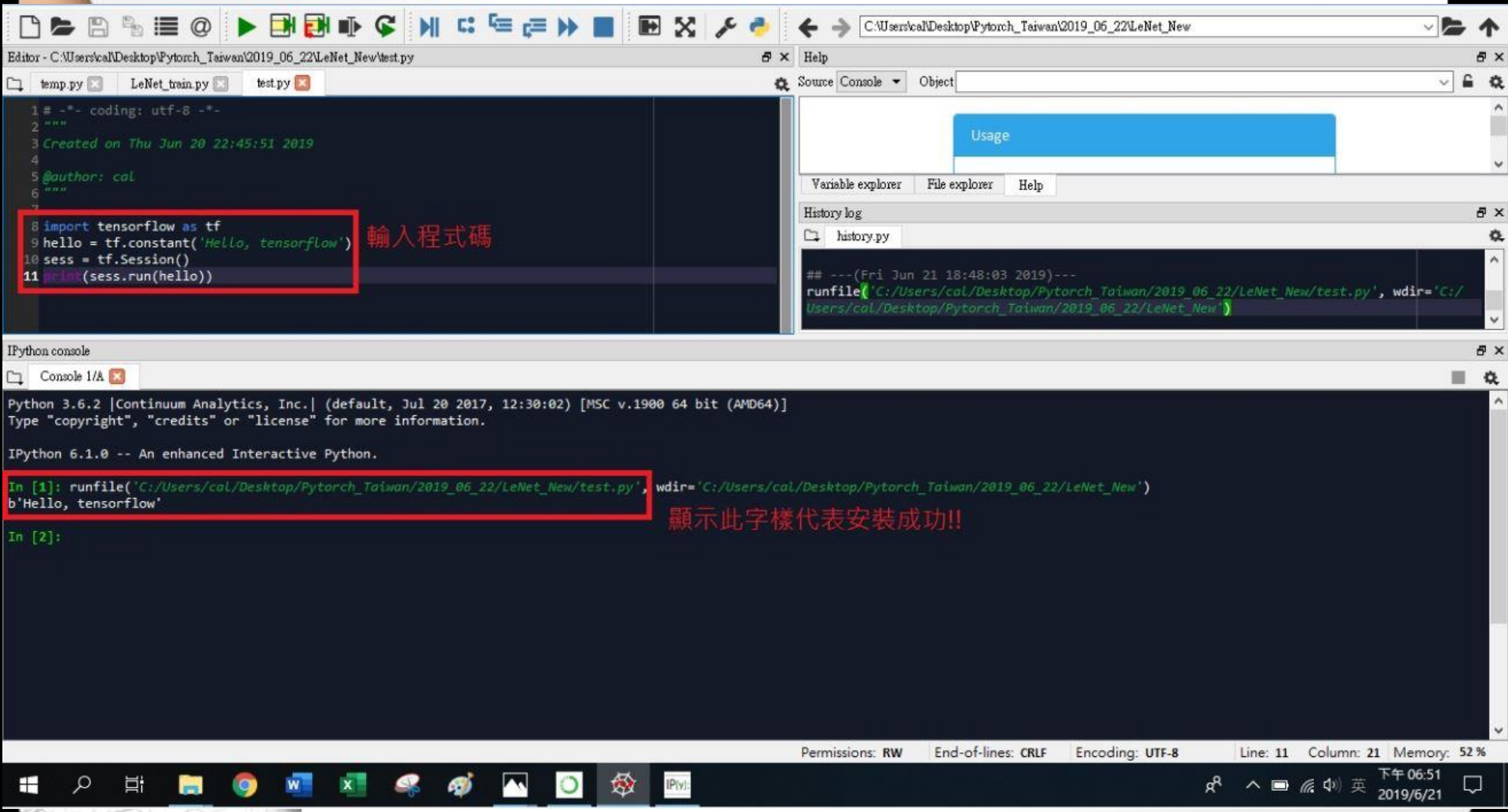
安装pip和cpu版tensorflow

```
(C:\Users\cal\Anaconda3\envs\tensorflow_cpu) C:\Users\cal>conda install -c anaconda pip
Fetching package metadata .............
Solving package specifications: .
```

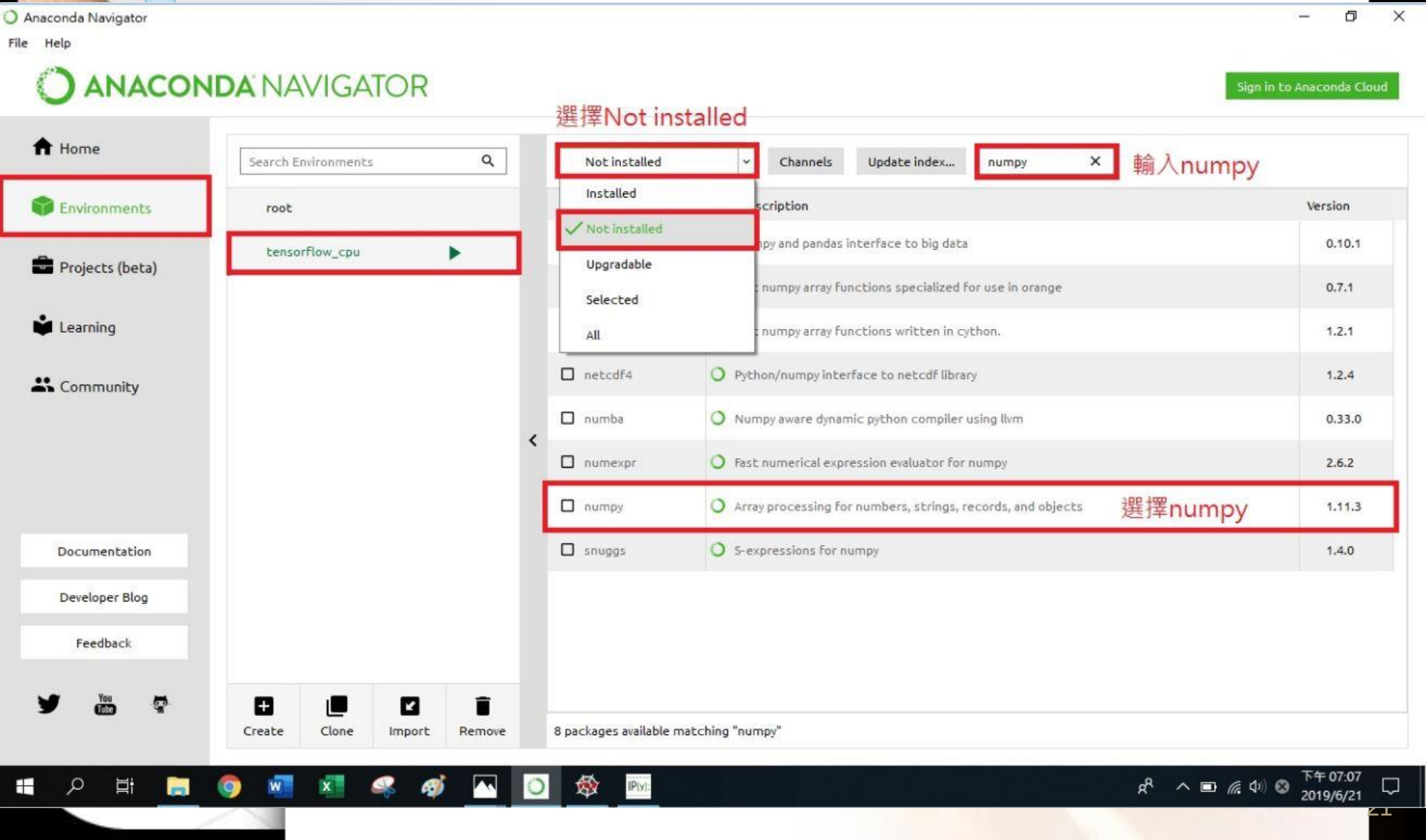
```
C:\Users\chinja>pip install tensorflow == 1.13.1
```

安裝Spyder


ANACONDA NAVIGATOR
Sign in to Anaconda Cloud
Home
Environments
Projects (beta)
Learning
Community
Documentation
Developer Blog
Feedback
Applications on
tensorflow_cpu
Channels
Refresh
glueviz
0.10.4
Multidimensional data visualization across files. Explore relationships within and among related datasets.
Install
notebook
5.0.0
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.
Install
orange3
3.4.1
Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.
Install
qtconsole
4.3.1
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.
Install
rstudio
1.1.456
A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.
Install
spyder
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features
Install
Install application spyder==3.2.3 on C:\Users\cal\Anaconda3\envs\tensorflow_cpu Fetching spyder-3.2.3-p...


C:\Users\cal\Desktop\Pytorch_Taiwan\2019_06_22\LeNet_New
Editor - C:\Users\cal\Desktop\Pytorch_Taiwan\2019_06_22\LeNet_New\test.py
temp.py
LeNet_train.py
test.py
-*- coding: utf-8 -*-
"""
Created on Thu Jun 20 22:45:51 2019
@author: cal
"""
import tensorflow as tf
hello = tf.constant('Hello, tensorflow')
sess = tf.Session()
print(sess.run(hello))
輸入程式碼
Help
Source
Console
Object
Usage
Variable explorer
File explorer
Help
History log
history.py
---(Fri Jun 21 18:48:03 2019)---
runfile('C:/Users/cal/Desktop/Pytorch_Taiwan/2019_06_22/LeNet_New/test.py', wdir='C:/Users/cal/Desktop/Pytorch_Taiwan/2019_06_22/LeNet_New')
IPython console
Console 1/A
Python 3.6.2 |Continuum Analytics, Inc.| (default, Jul 20 2017, 12:30:02) [MSC v.1900 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.
IPython 6.1.0 -- An enhanced Interactive Python.
In [1]: runfile('C:/Users/cal/Desktop/Pytorch_Taiwan/2019_06_22/LeNet_New/test.py', wdir='C:/Users/cal/Desktop/Pytorch_Taiwan/2019_06_22/LeNet_New')
b'Hello, tensorflow'
In [2]:
顯示此字樣代表安裝成功!!
Permissions: RW
End-of-lines: CRLF
Encoding: UTF-8
Line: 11
Column: 21
Memory: 52 %
下午 06:51
2019/6/21

Numpy安裝

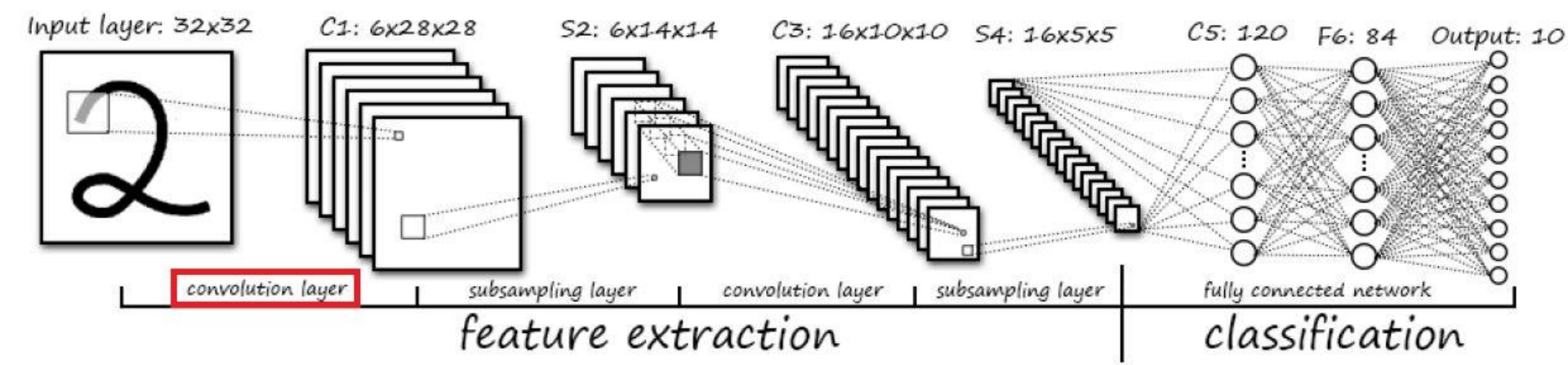

Anaconda Navigator
File Help
ANACONDA NAVIGATOR
Sign in to Anaconda Cloud
選擇Not installed
Home
Environments
Projects (beta)
Learning
Community
Search Environments
root
tensorflow_cpu
Not installed
Channels
Update index...
numpy
輸入numpy
Installed
Not installed
Upgradable
Selected
All
Version
0.10.1
0.7.1
1.2.1
netcdf4
Python/numpy interface to netcdf library
1.2.4
numba
Numpy aware dynamic python compiler using llvm
0.33.0
numexpr
Fast numerical expression evaluator for numpy
2.6.2
numpy
Array processing for numbers, strings, records, and objects
選擇numpy
1.11.3
snuggs
S-expressions for numpy
1.4.0
Documentation
Developer Blog
Feedback
Create
Clone
Import
Remove
8 packages available matching "numpy"
下午 07:07
2019/6/21

Tensorflow版

Layer 1: Convolution

```
#Layer 1: Convolution (32x32x1 -> 28x28x6)
conv1_w = tf.Variable(tf.random_normal([5,5,1,6]))
conv1_b = tf.Variable(tf.zeros([6]))
conv1 = tf.nn.conv2d(x, conv1_w, strides = [1,1,1,1], padding = 'VALID') + conv1_b
## Activation function
conv1 = tf.nn.sigmoid(conv1)
```

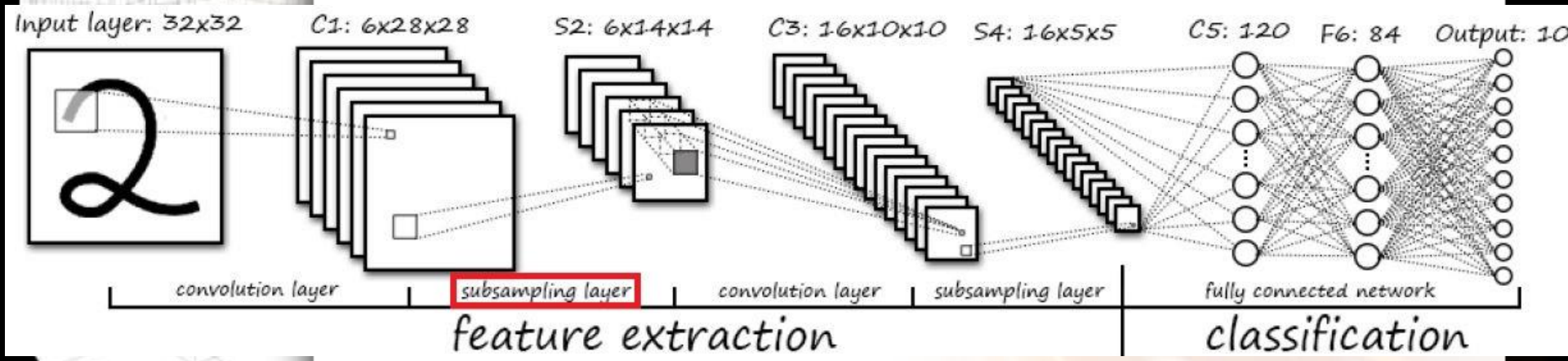
Layer C1 is a convolutional layer with 6 feature maps. Each unit in each feature map is connected to a 5x5 neighborhood in the input. The size of the feature maps is 28x28



Layer 2: Max pooling

```
#Layer 2: Max pooling (28x28x6 -> 14x14x6)
conv1 = tf.nn.max_pool(conv1, ksize = [1,2,2,1], strides = [1,2,2,1], padding = 'VALID')
```

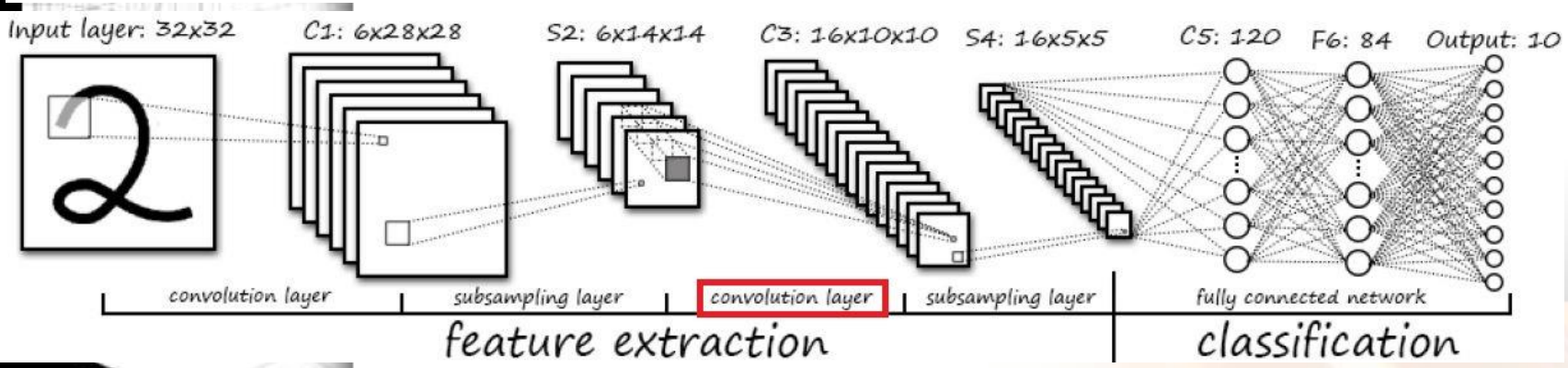
Layer S2 is a sub-sampling layer with 6 feature maps of size 14x14. Each unit in each feature map is connected to a 2x2 neighborhood in the corresponding feature map in C1.



Layer 3: Convolution

```
#Layer 3: Convolution (14x14x6 -> 10x10x16)
conv2_w = tf.Variable(tf.random_normal([5,5,6,16]))
conv2_b = tf.Variable(tf.zeros([16]))
conv2 = tf.nn.conv2d(conv1, conv2_w, strides = [1,1,1,1], padding = 'VALID') + conv2_b
## Activation function
conv2 = tf.nn.sigmoid(conv2)
```

Layer C3 is a convolutional layer with 16 feature maps. Each unit in each feature map is connected to several 5x5 neighborhoods at identical locations in a subset of S2's feature maps. Table I shows the set of S2 feature maps



Layer C3:

1516 trainable parameters.

=(3*5*5+1)*6+(4*5*5+1)*9+(6*5*5+1)

Connections: 151600

(3*5*5+1)*6*10*10+(4*5*5+1)*9*10*10
+(6*5*5+1)*10*10

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X				X	X	X			X	X	X	X		X	X
1	X	X				X	X	X			X	X	X	X		X
2	X	X	X				X	X	X			X		X	X	X
3		X	X	X			X	X	X	X			X		X	X
4			X	X	X			X	X	X	X		X	X		X
5				X	X	X			X	X	X	X		X	X	X

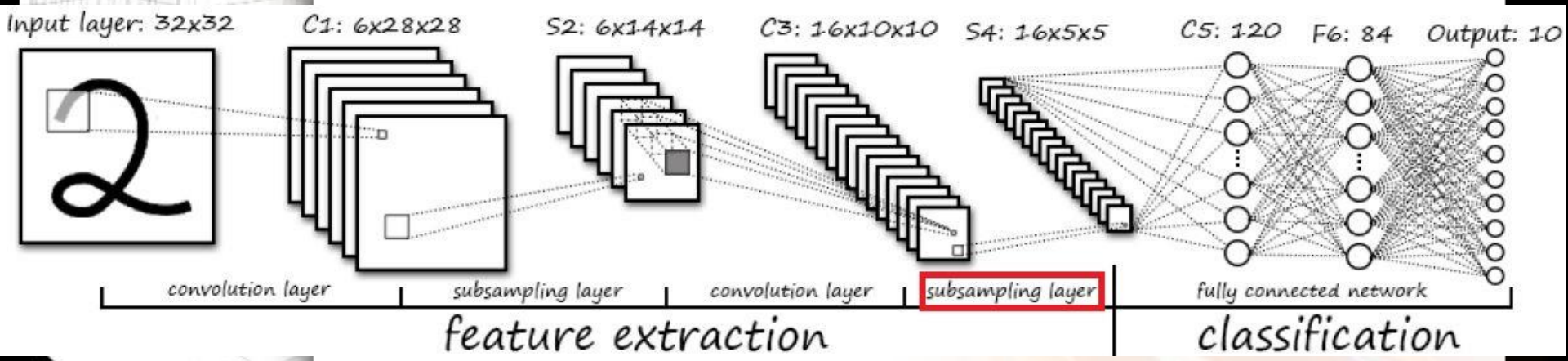
TABLE I

EACH COLUMN INDICATES WHICH FEATURE MAP IN S2 ARE COMBINED BY THE UNITS IN A PARTICULAR FEATURE MAP OF C3.

Layer 4: Max pooling

```
#Layer 4: Max pooling (10x10x16 -> 5x5x16)
conv2 = tf.nn.max_pool(conv2, ksize = [1,2,2,1], strides = [1,2,2,1], padding = 'VALID')
```

Layer S4 is a sub-sampling layer with 16 feature maps of size 5x5. Each unit in each feature map is connected to a 2x2 neighborhood in the corresponding feature map in C3, in a similar way as C1 and S2. Layer S4 has 32 trainable

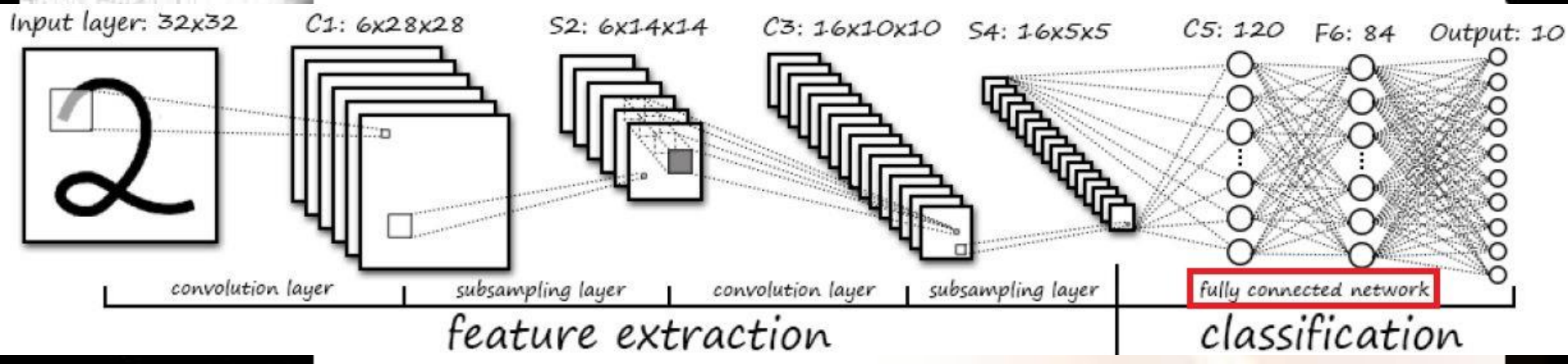


Layer 5: Fully Connected

```
#Layer 5: Fully Connected (5x5x16 = 400 -> 120)
dense1_w = tf.Variable(tf.random_normal([400,120]))
dense1_b = tf.Variable(tf.zeros([120]))
conv2 = flatten(conv2)
dense1 = tf.matmul(conv2, dense1_w) + dense1_b
## Activation function
dense1 = tf.nn.sigmoid(dense1)
```

Layer C5 is a convolutional layer with 120 feature maps. Each unit is connected to a 5x5 neighborhood on all 16 of S4's feature maps. Here, because the size of S4 is also 5x5, the size of C5's feature maps is 1x1: this amounts to a full connection between S4 and C5. C5 is labeled

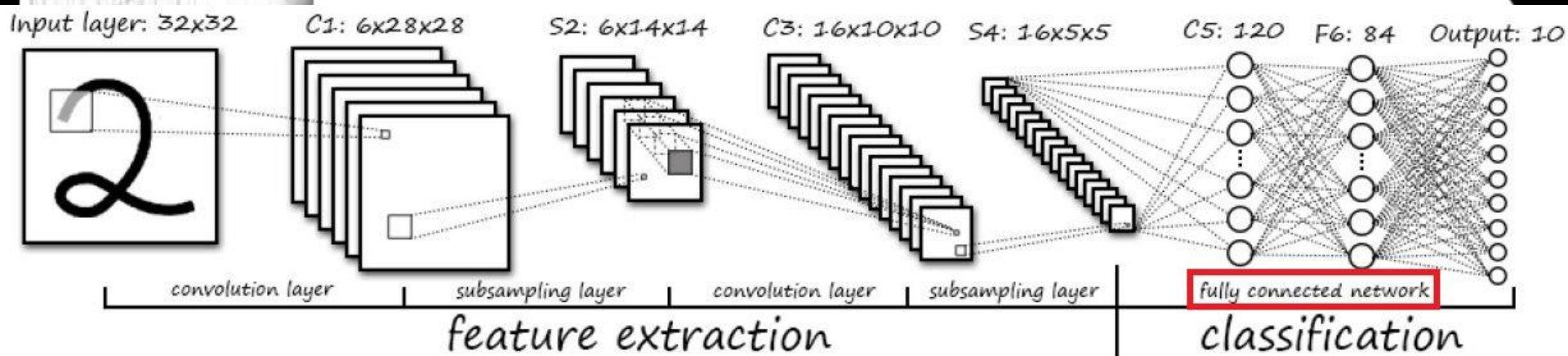
湊巧變成FC!!!



Layer 6: Fully Connected

```
#Layer 6: Fully Connected (120 -> 84)
dense2_w = tf.Variable(tf.random_normal([120,84]))
dense2_b = tf.Variable(tf.zeros([84]))
dense2 = tf.matmul(dense1, dense2_w) + dense2_b
## Activation function
dense2 = tf.nn.sigmoid(dense2)
```

Layer F6, contains 84 units (the reason for this number comes from the design of the output layer, explained below) and is fully connected to C5. It has 10,164 trainable parameters.



Layer 7: Fully Connected

```
#Layer 7: Fully Connected (84 -> 10)
dense3_w = tf.Variable(tf.random_normal([84,10]))
dense3_b = tf.Variable(tf.zeros([10]))
dense3 = tf.matmul(dense2, dense3_w) + dense3_b

return dense3
```

In other words, each output RBF unit computes the Euclidean distance between its input vector and its parameter vector. The further away is the input from the parameter vector, the larger is the RBF output. The output of a 7x12 bitmap (hence the number 84)

Output layer: 10RBF (One for each digit)

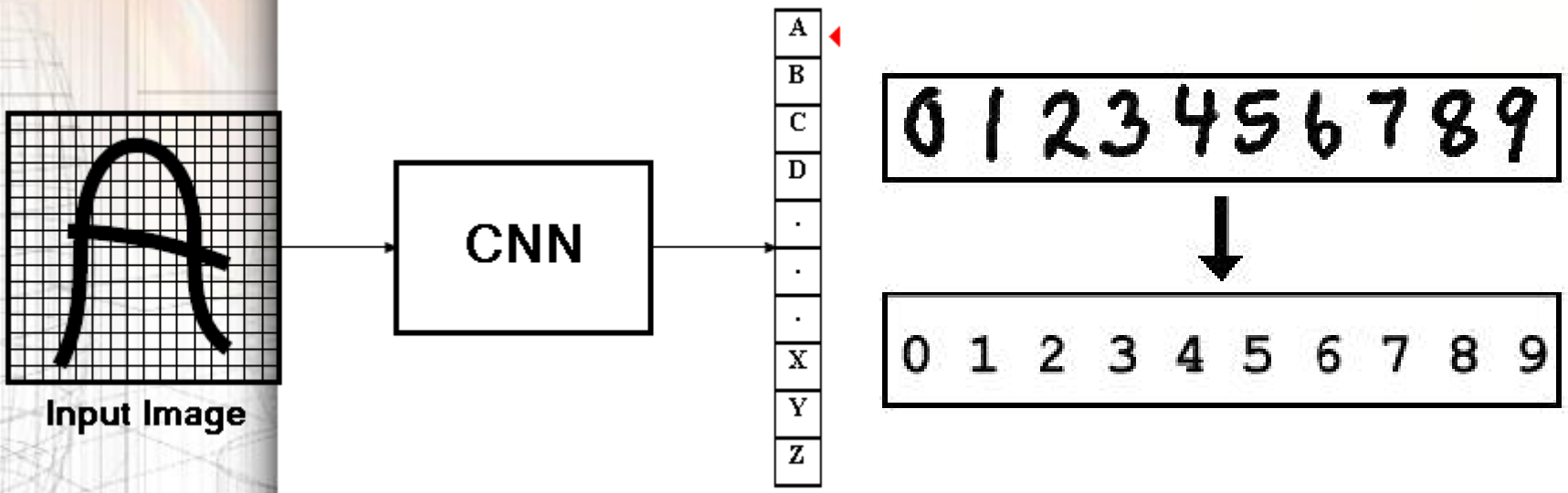
$$y_i = \sum_{j=1}^{84} (x_j - w_{ij})^2, \ i = 1, \ldots, 10.$$

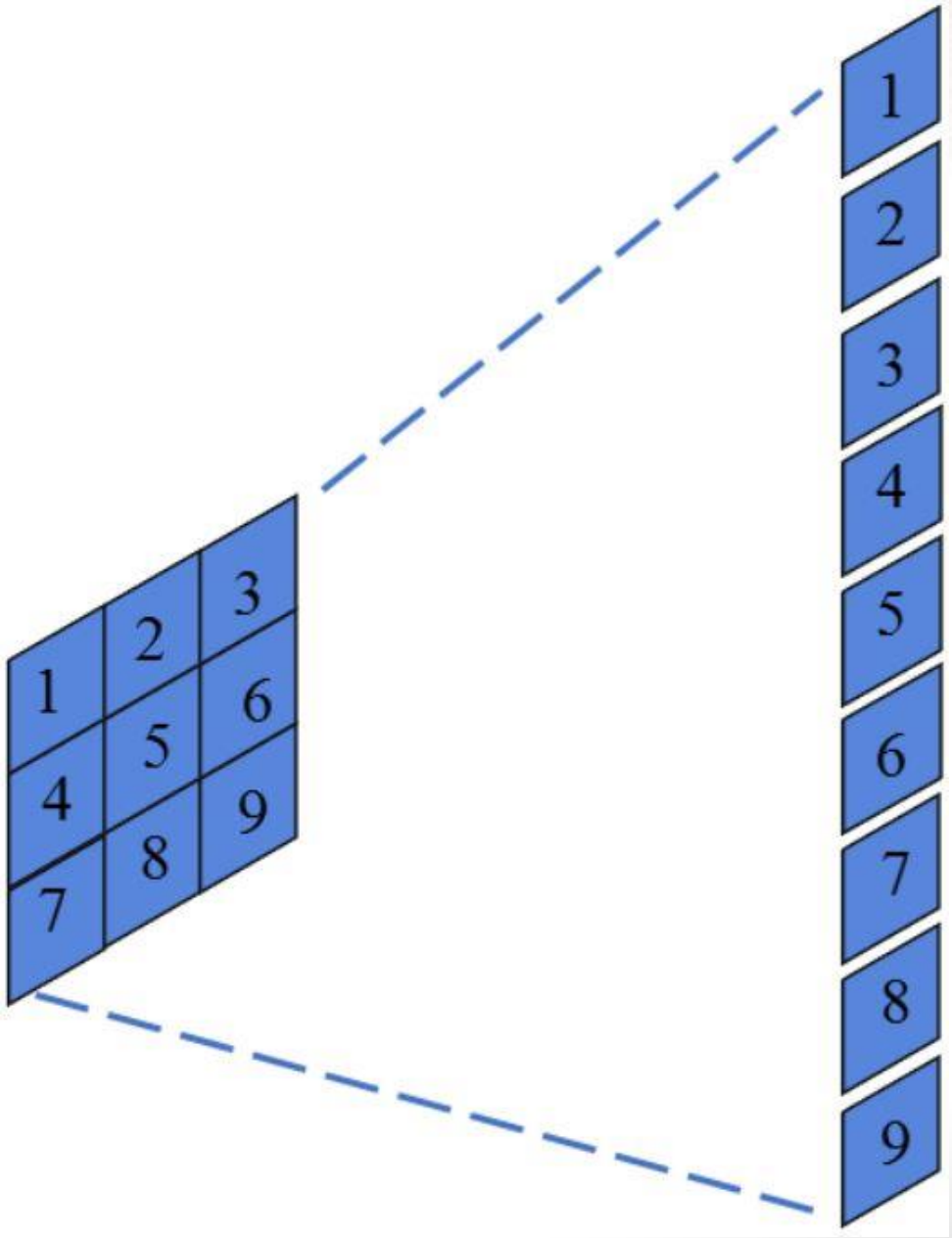
From F6

84=7x12, stylized image.

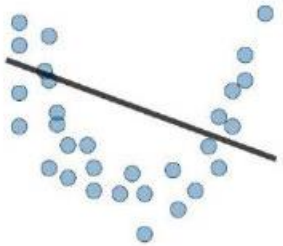
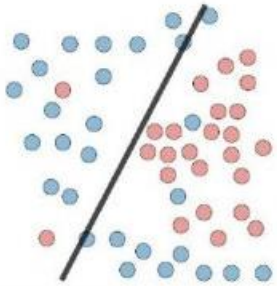
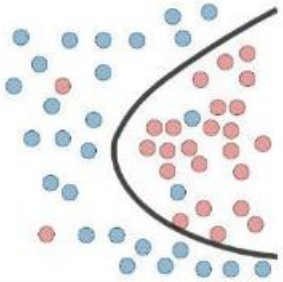
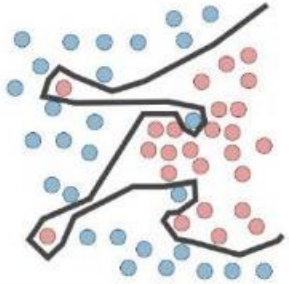
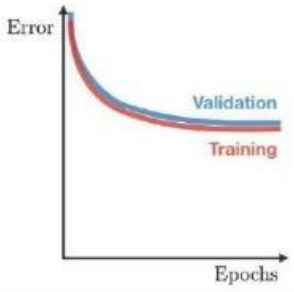
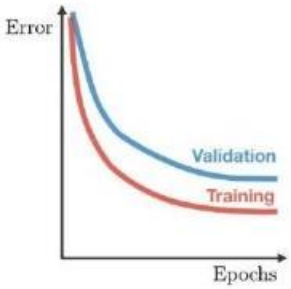
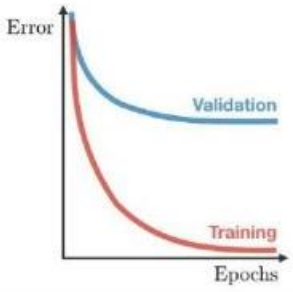
84 parameters, 84*10 connections

Fully Connected



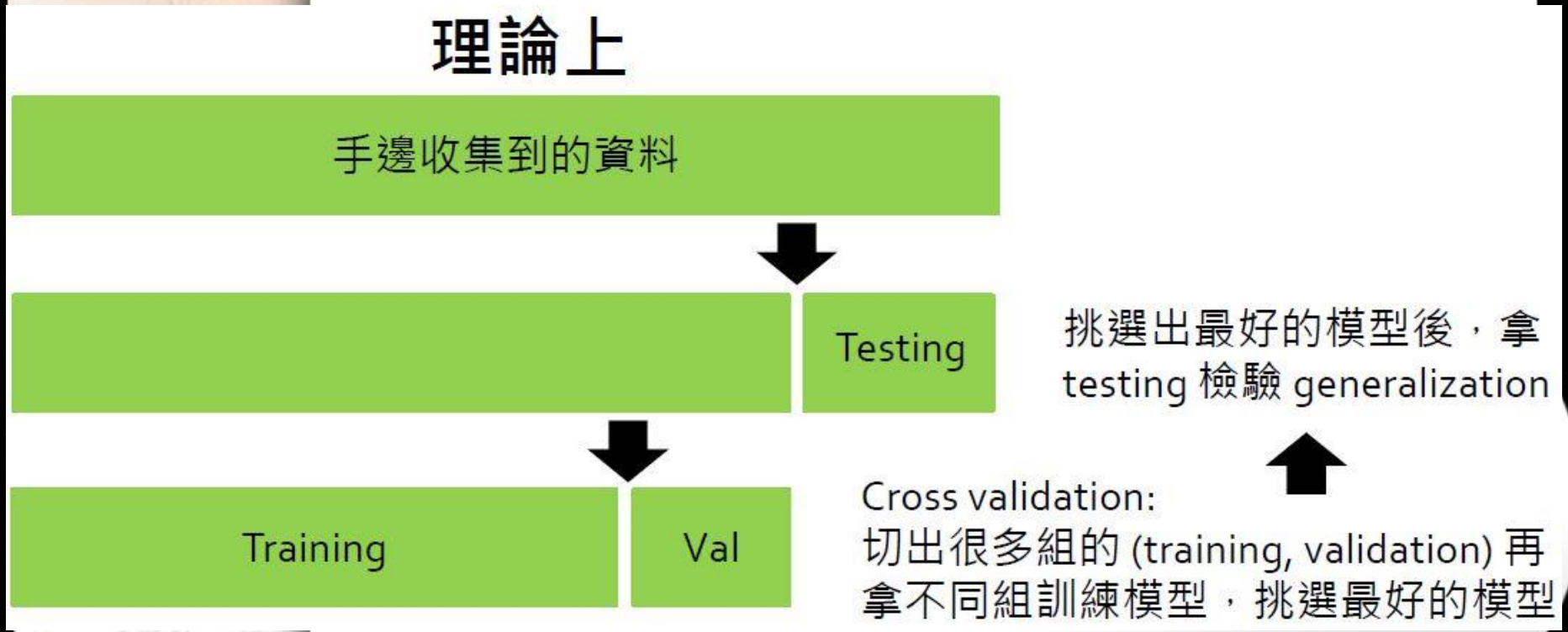

1
2
3
4
5
6
7
8
9
1
2
3
4
5
6
7
8
9

補充

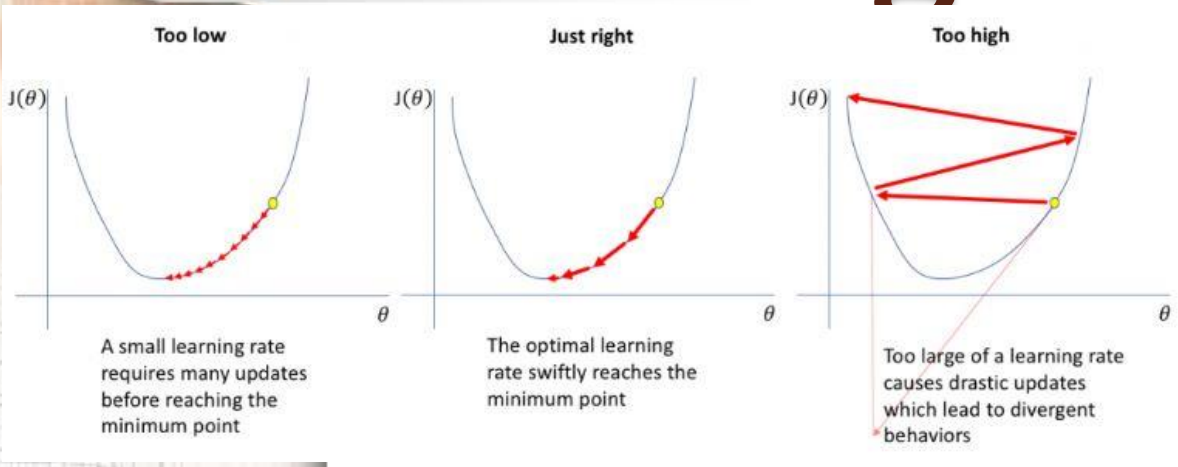
	Underfitting	Just right	Overfitting
Symptoms	- High training error - Training error close to test error - High bias	- Training error slightly lower than test error	- Low training error - Training error much lower than test error - High variance
Regression			
Classification			
Deep learning			
Remedies	- Complexify model - Add more features - Train longer		- Regularize - Get more data

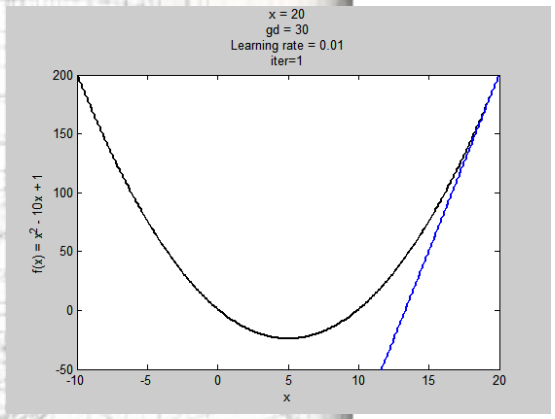
Validation Dataset

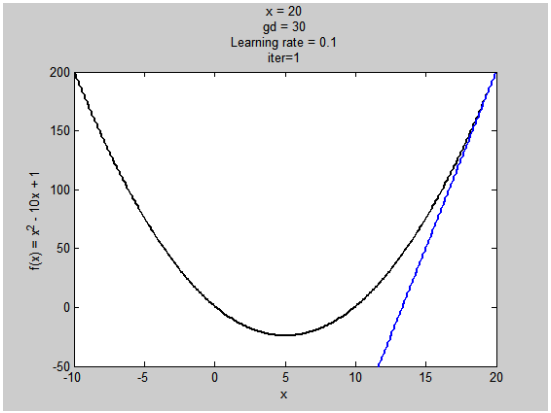
- **Validation dataset** 用來挑選模型
- **Testingdataset**檢驗模型的**普遍性(generalization)**
  避免模型過度學習training dataset

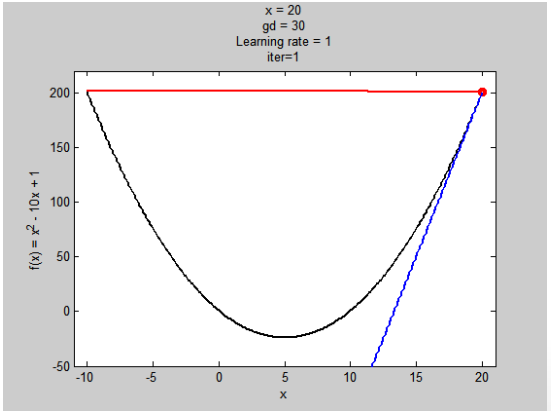


Learning Rate



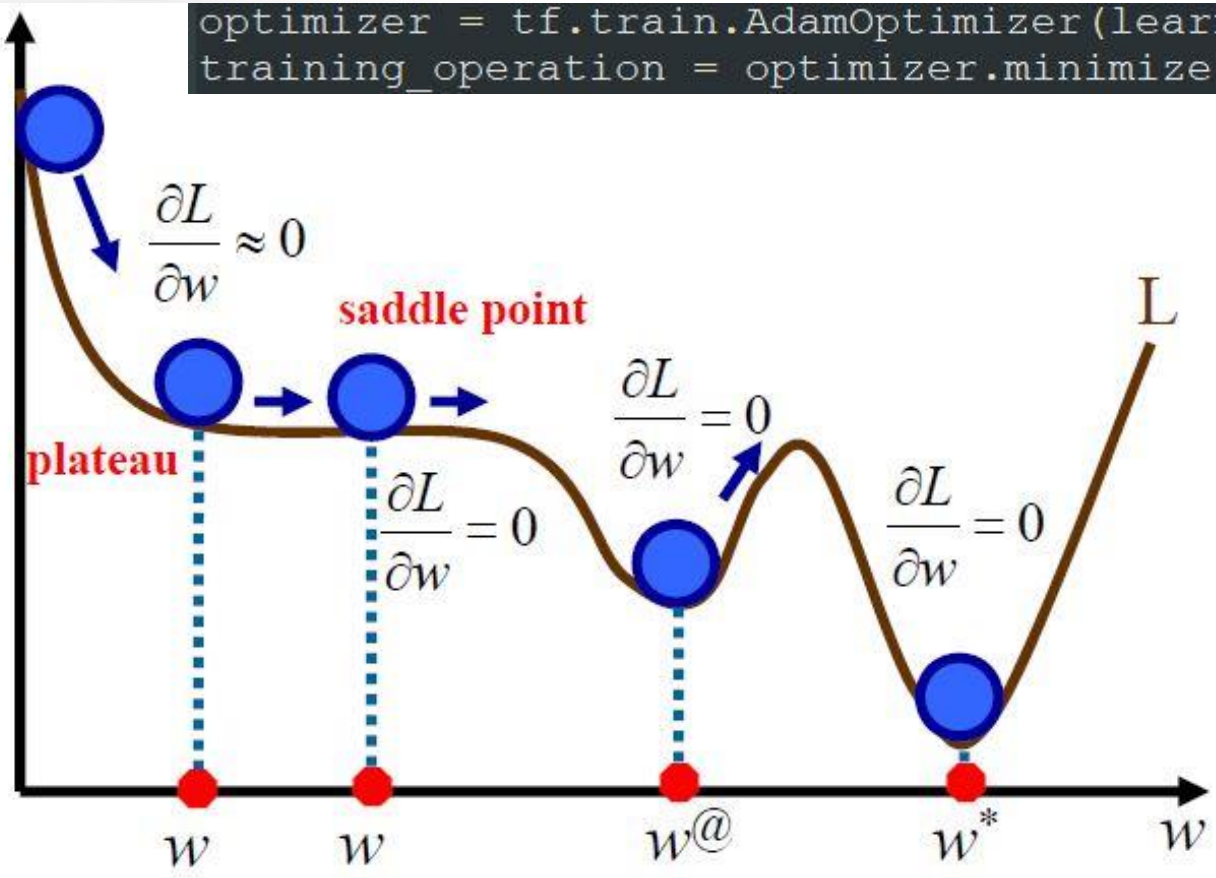






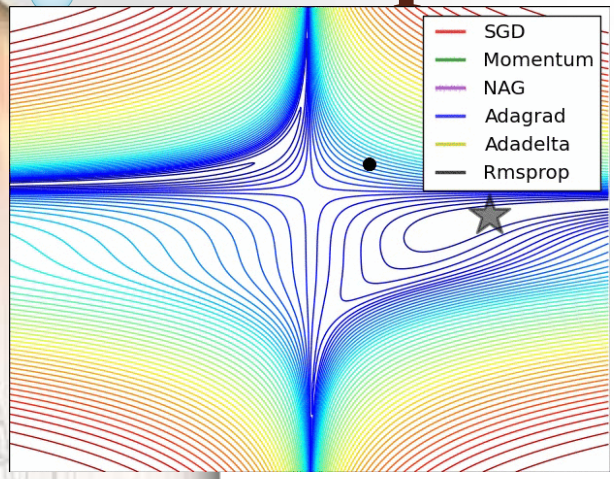
Optimizer

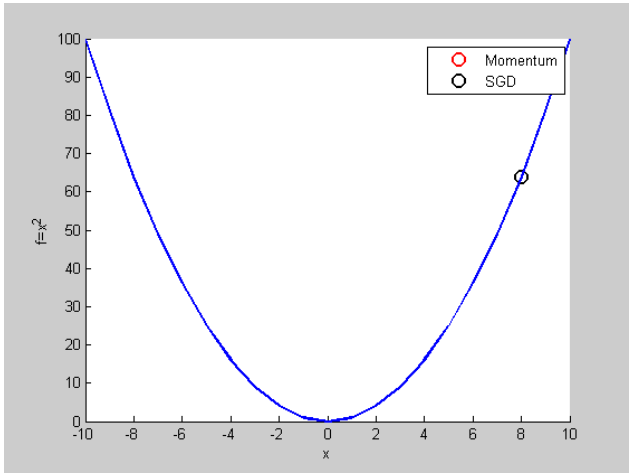
```
optimizer = tf.train.AdamOptimizer(learning_rate = rate)
training_operation = optimizer.minimize(loss_operation)
```

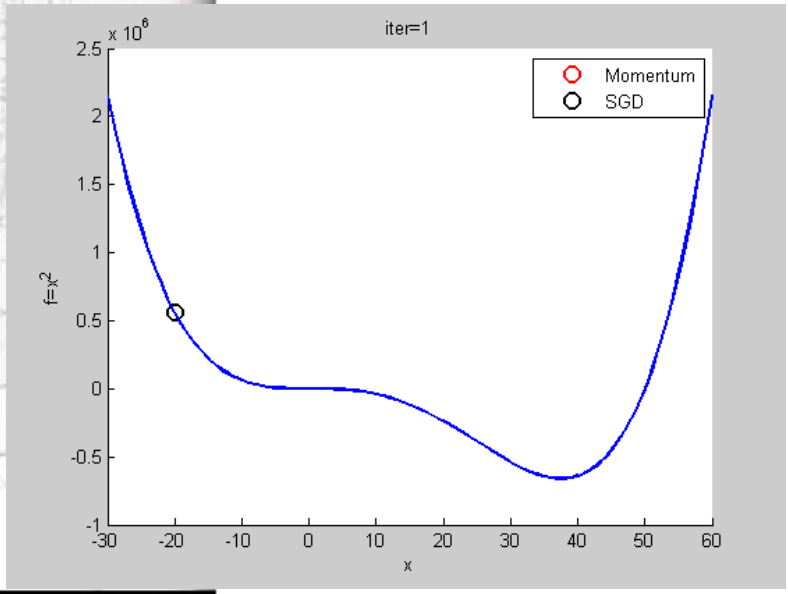


Local Optimal **Global Optimal**

SGD, Momentum, AdaGrad, Adam Optimizer







TAIPEI TECH

純Python版

1_Layer

```
1  from numpy import *
2
3  #建立 [Size x Size] x Numbers Neurons 的 Layer
4  class Layer(object):
5      def __init__(self, layer_size = []):
6          self.layer_size = layer_size
7          self.maps = []
8
9          for map_size in layer_size :
10             self.maps.append(zeros(map_size))
11         self.maps = array(self.maps)
```

2_CovLayer

```
18 class CovLayer(Layer):
19     def __init__(self, layer_size = [], cov_core_sizes = [], mapcombindex = []):
20         Layer.__init__(self, layer_size)
21         self.covcores = []
22         self.covbias = []
23         self.mapcombindex = mapcombindex
24 
25         #初始化Parameters.
26         #-2.4/Fi 和 2.4/Fi 的值都來自於論文的 Appendices A.
27         for cov_core_size in cov_core_sizes:
28             #Fi is from the definition in paper
29             Fi = cov_core_size[0] * cov_core_size[1] + 1
30             #Make random filters
31             self.covcores.append(random.uniform(-2.4/Fi, 2.4/Fi, cov_core_size))
32             #Make random biases
33             self.covbias.append(random.uniform(-2.4/Fi, 2.4/Fi))#卷積的bias
34 
35         self.covcores = array(self.covcores)
```

Before training, the weights are initialized with random values using a uniform distribution between $-2.4/F_i$ and $2.4/F_i$ where F_i is the number of inputs (fan-in) of the unit which the connection belongs to. Since several connections

```
37      def cov_op(self, pre_maps, covcore_index):
38          pre_map_shape = pre_maps.shape
39          covcore_shape = self.covcores[covcore_index].shape
40          map_shape = self.maps[covcore_index].shape
41      #檢查輸入size
42      #如果輸入size和輸出size相符合
43      #計算卷積層
44          if not(map_shape[-2] == pre_map_shape[-2] - covcore_shape[-2] + 1 \
45             and map_shape[-1] == pre_map_shape[-1] - covcore_shape[-2] + 1):
46              return None;
47          for i in range(map_shape[-2]):
48              for j in range(map_shape[-1]):
49                  #Filter caculation
50                  localrecept = pre_maps[:, i : i + covcore_shape[-2], j : j + covcore_shape[-1]]
51                  val = sum(localrecept * self.covcores[covcore_index]) + self.covbias[covcore_index
52
53                  #Use tanh(x) as activation function.
54                  #Remark that tanh(x) = ((e^2x)/(e^2x)+1)
55                  #We use the parameters in the paper
56                  #f(a) = Atanh(Sa), where A = 1.7159 * (val - 1)/(val + 1)
57
58                  val = exp((4.0/3)*val)
59                  self.maps[covcore_index][i][j] = 1.7159 * (val - 1) / (val + 1)
```

directions [116]. For our simulations, we use $A = 1.7159$ and $S = \frac{2}{3}$ (see [20], [34]). With this choice of parameters, the equalities $f(1) = 1$ and $f(-1) = -1$ are satisfied.

```
def back_propagation(self, pre_mapset, current_error, learn_rate, isweight_update):
    self.current_error = current_error

    #Flatten the array into one dim(把array拉成一維vector)
    selfmap_line = self.maps.reshape([self.maps.shape[0] * self.maps.shape[1] * self.maps.shape[2]])
    currenterror_line = current_error.reshape([current_error.shape[0] * current_error.shape[1] * current_error.shape[2]])

    # Backward : df/da = 2/3 * 1.7159 * (1-(tanh(2/3*a)^2))
    # Where tanh(2/3*a) = selfmap_line[i]/1.7159
    pcurrent_error = array([((2.0/3)*(1.7159 - (1/1.7159) * selfmap_line[i]**2))*currenterror_line[i]\
                            for i in range(len(selfmap_line))]).reshape(self.maps.shape)
```

directions [116]. For our simulations, we use $A = 1.7159$ and $S = \frac{2}{3}$ (see [20], [34]). With this choice of parameters, the equalities $f(1) = 1$ and $f(-1) = -1$ are satisfied.

3_Pooling_Layer

10	15	80	12
18	30	11	20
11	19	48	51
30	31	45	59

max(10,15,18,30)=30	max(80,12,11,20)=80
max(11,19,30,31)=31	max(48,51,45,59)=59

30	80
31	59

```
10 class PoolingLayer(Layer) :
11         def __init__(self, lay_size = [], pool_core_sizes = []) :
12                 Layer.__init__(self, lay_size)
13                 Fi = pool_core_sizes[0][0] * pool_core_sizes[0][1] + 1
14                 self.poolparas = random.uniform(-2.4/Fi, 2.4/Fi, [len(lay_size), 2])
15                 self.poolcore_sizes = array(pool_core_sizes)
16
17         def pool_op(self, pre_map, pool_index) :
18                 pre_map_shape = pre_map.shape
19                 poolcore_size = self.poolcore_sizes[pool_index]
20                 for i in range(int(pre_map_shape[0] / poolcore_size[0])) :
21                         for j in range(int(pre_map_shape[1] / poolcore_size[1])) :
22                                 val = self.poolparas[pool_index][0] * sum(pre_map[i*poolcore_size[0]:(i+1)*poolcore_size[0],\
23                                     j*poolcore_size[1]:(j+1)*poolcore_size[1]]) + self.poolparas[pool_index][1]
24                                 val = exp((4.0/3)*val)
25                                 self.maps[pool_index][i][j] = 1.7159 * (val -1) / (val + 1)
```

4_FullyConLayer

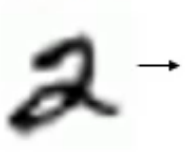
```
11  class FcLayer(Layer) :
12          def __init__(self, layer_len, pre_nodesnum) :
13                  Layer.__init__(self, [[1, layer_len]])
14                  Fi = pre_nodesnum + 1
15                  self.weight = random.uniform(-2.4/Fi, 2.4/Fi, [layer_len, pre_nodesnum]) # 84x120
16                  self.bias = random.uniform(-2.4/Fi, 2.4/Fi, [layer_len]) #84
17
18                  val  = sum(self.weight[node_index] * pre_nodes) + self.bias[node_index] #
19                  val = exp((4.0/3)*val)
20                  self.maps[0][0][node_index] = 1.7159 *  (val -1) / (val + 1)
21
22          def calc_maps(self, pre_mapset) :
23                  for i in range(len(self.maps[0][0])) : #84
24                          self.fc_op(pre_mapset, i)
```

```
29          def back_propagation(self, pre_mapset, current_error, learn_rate, isweight_update) :
30              self.current_error = current_error
31              pcurrent_error = [((2.0/3)*(1.7159 - (1/1.7159) * self.maps[0][0][i]**2))*current_error[0][0][i]\
32                  for i in range(self.maps.shape[-1])]
33
34              # 1x84 > 84x1 dot 1x120 > 84x120
35              weight_update = dot(matrix(pcurrent_error).T, \
36                  matrix(pre_mapset.reshape([1, pre_mapset.shape[0] * pre_mapset.shape[1] * pre_mapset.shape[2]])))
37
38              bias_update = array(pcurrent_error)
39
40              if isweight_update :
41                  self.weight -= learn_rate * weight_update
42                  self.bias -= learn_rate * bias_update
43              pre_error = array(dot(matrix(pcurrent_error), matrix(self.weight))).reshape(pre_mapset.shape)
44              return pre_error
```

_6_ConNet

```
14  class CovNet(object) :
15      def __init__(self) :
16
17          cov3_core_sizes = [[3, 5, 5]] * 6
18          cov3_core_sizes.extend([[4, 5, 5]] * 9)
19          cov3_core_sizes.extend([[6, 5, 5]])
20
21          cov3_mapcombindex = [[0,1,2],[1,2,3],[2,3,4],[3,4,5],[4,5,0],[5,0,1],\
22                  [0,1,2,3],[1,2,3,4],[2,3,4,5],[3,4,5,0],[4,5,0,1],[5,0,1,2],[0,1,3,4],[1,2,4,5],[0,2,3,5],[0,1,2,3,4,5]]
23
24          self.covlay1 = CovLayer([[28, 28]] * 6, [[1, 5, 5]] * 6)
25          self.poolinglay2 = PoolingLayer([[14, 14]] * 6, [[2, 2]] * 6)
26
27          self.covlay3 = CovLayer([[10, 10]] * 16, cov3_core_sizes, cov3_mapcombindex)
28          self.poolinglay4 = PoolingLayer([[5, 5]] * 16, [[2, 2]] * 16)
29
30          self.covlay5 = CovLayer([[1, 1]] * 120, [[16, 5, 5]] * 120)
31          self.fclay6 = FcLayer(84, 120)
32
33          self.outputlay7 = OutputLayer(10, 84)
```

Input image



32 x 32 x 1

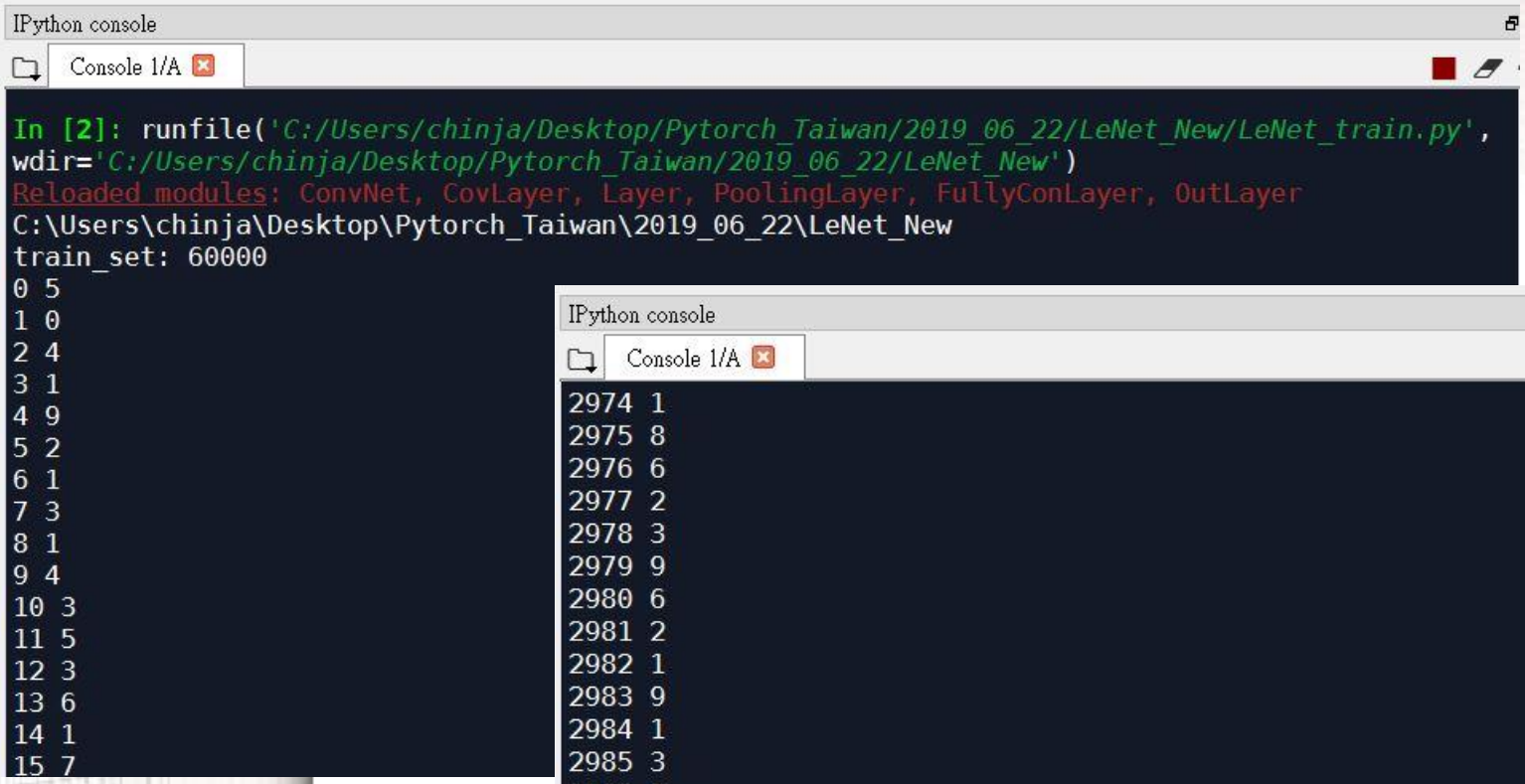
_7_Net_TrainAndTest

```
14  #mnist has a training set of 60,000 examples, and a test set of 10,000 examples.
15  #log檔作用:紀錄檔案（logfile）是一個記錄了發生在執行中的作業系統或其他軟體中的事件的檔案
16  def train_net(train_covnet, logfile, cycle, learn_rate, case_num = -1) :
17      # Read data
18      # Change it to your own dataset path
19      trainim_filepath = '../data/raw/train-images.idx3-ubyte' #training的資料(按照自己的路徑修改)
20      trainlabel_filepath = '../data/raw/train-labels.idx1-ubyte' #label的資料(按照自己的路徑修改)
21
22      trainimfile = open(trainim_filepath, 'rb') #open()開啟檔案
23      trainlabelfile = open(trainlabel_filepath, 'rb') #使用'rb'按照二進位制位進行讀取的，不會將讀取的位元組轉換成字元
24
25      train_im = trainimfile.read() # 讀取文件內容 f.read(size) - 回傳檔案內容,
26      train_label = trainlabelfile.read() #size為要讀取進來的字串長度，若不填則讀取整份文件
27      im_index = 0
28      label_index = 0
29      magic, numImages , numRows , numColumns = struct.unpack_from('>IIII' , train_im , im_index)
30      magic, numLabels = struct.unpack_from('>II', train_label, label_index)
31      print ('train_set:', numImages)
32
33      train_btime = time.time()
34      logfile.write('learn_rate:' + str(learn_rate) + '\t')
35      logfile.write('train_cycle:' + str(cycle) + '\t')
```

```
    # Begin to train
    for c in range(cycle) :
        im_index = struct.calcsize('>IIII')
        label_index = struct.calcsize('>II')
        train_case_num = numImages
        if case_num != -1 and case_num < numImages :
            train_case_num = case_num
        logfile.write("trainset_num:" + str(train_case_num) + '\t')
        for case in range(train_case_num) :
            im = struct.unpack_from('>784B', train_im, im_index)
            label = struct.unpack_from('>1B', train_label, label_index)
            im_index += struct.calcsize('>784B')
            label_index += struct.calcsize('>1B')
            im = array(im)
            im = im.reshape(28,28)
            bigim = list(ones((32, 32)) * -0.1)
            for i in range(28) :
                for j in range(28) :
                    if im[i][j] > 0 :
                        bigim[i+2][j+2] = 1.175
            im = array([bigim])
            label = label[0]
            print (case, label)
            train_covnet.fw_prop(im, label)
            train_covnet.bw_prop(im, label, learn_rate[c])
```

```
    # 測試
    for case in range(testcase_num) :
        im = struct.unpack_from('>784B', test_im, im_index)
        label = struct.unpack_from('>1B', test_label, label_index)
        im_index += struct.calcsize('>784B')
        label_index += struct.calcsize('>1B')
        im = array(im)
        im = im.reshape(28,28)
        bigim = list(ones((32, 32)) * -0.1)
        for i in range(28) :
            for j in range(28) :
                if im[i][j] > 0 :
                    bigim[i+2][j+2] = 1.175
        im = array([bigim])
        label = label[0]
        print( case, label)
        train_covnet.fw_prop(im)
        if argmax(train_covnet.outputlay7.maps[0][0]) == label :
            correct_num += 1
    correct_rate = correct_num / float(testcase_num)
    print('test_correct_rate:', correct_rate)
    logfile.write('test_correct_rate:'+ str(correct_rate) + '\t')
    logfile.write('\n')
```

結果


IPython console
Console 1/A
In [2]: runfile('C:/Users/chinja/Desktop/Pytorch_Taiwan/2019_06_22/LeNet_New/LeNet_train.py',
wdir='C:/Users/chinja/Desktop/Pytorch_Taiwan/2019_06_22/LeNet_New')
Reloaded modules: ConvNet, CovLayer, Layer, PoolingLayer, FullyConLayer, OutLayer
C:\Users\chinja\Desktop\Pytorch_Taiwan\2019_06_22\LeNet_New
train_set: 60000
0 5
1 0
2 4
3 1
4 9
5 2
6 1
7 3
8 1
9 4
10 3
11 5
12 3
13 6
14 1
15 7
IPython console
Console 1/A
2974 1
2975 8
2976 6
2977 2
2978 3
2979 9
2980 6
2981 2
2982 1
2983 9
2984 1
2985 3
2986 5
2987 5
2988 0
2989 3
2990 8
2991 3
2992 3
2993 7
2994 6
2995 6
2996 0
2997 1
2998 4
2999 0
test_correct_rate: 0.9336666666666666
In [4]:

TAIPEI TECH

感謝聆聽！