Python programming Basics

Simple MNIST convnet

Author: fchollet
Date created: 2015/06/19
Last modified: 2020/04/21
Description: A simple convnet that achieves ~99% test accuracy on MNIST.

 **View in Colab** •  **GitHub source**

MLP_neupy [~/Desktop/py_code_2020/MLP_neupy] - .../cnn_mnist.py [MLP_neupy]

LP_neupy 〉 cnn_mnist.py

cnn_mnist

Pr...

mnist_mlp.py | machine_translator.py | cnn_mnist.py | base.py | load_data_show.py

MLP_neupy ~/Deskt
venv
cnn_mnist.py
load_data_show.py
machine_translator
mnist_mlp.py
External Libraries
Scratches and Console

```python
import numpy as np
from tensorflow import keras
from tensorflow.keras import layers
# Model / data parameters
num_classes = 10
input_shape = (28, 28, 1)

# Load the data and split it between train and test sets
(x_train, y_train), (x_test, y_test) = keras.datasets.mnist.l

# Scale images to the [0, 1] range
x_train = x_train.astype("float32") / 255
x_test = x_test.astype("float32") / 255
# Make sure images have shape (28, 28, 1)
x_train = np.expand_dims(x_train, -1)
x_test = np.expand_dims(x_test, -1)
```

```
Epoch 14/15
422/422 [==============================] - 17s 39ms/step - loss: 0.0348
 - accuracy: 0.9886 - val_loss: 0.0295 - val_accuracy: 0.9918
Epoch 15/15
422/422 [==============================] - 16s 38ms/step - loss: 0.0315
 - accuracy: 0.9895 - val_loss: 0.0303 - val_accuracy: 0.9915
Test loss: 0.026409737765789032
Test accuracy: 0.9909999966621399
```

IMAGENET影像辨識

IMAGENET

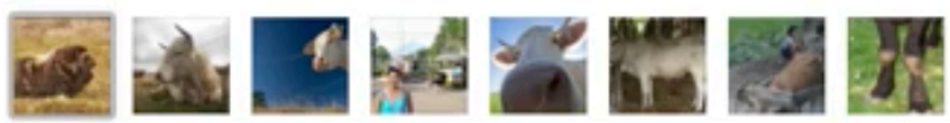
14,197,122 images, 21841 synsets indexed

Home Download Challenges About

Not logged in. Login I Signup

ImageNet is an image database organized according to the WordNet hierarchy (currently only the nouns), in which each node of the hierarchy is depicted by hundreds and thousands of images. The project has been instrumental in advancing computer vision and deep learning research. The data is available for free to researchers for non-commercial use.

Mar 11 2021. ImageNet website update.



list

使用方括號逗點分隔元素

```
list = ['Thomson', 78, 12.28, 'Sunny', 182.2]
tinylist = [123, 'Tony']
print("list:", list)
print("list[0]:", list[0])
print("list[1:3]:", list[1:3])
print("list[2:]:", list[2:])
print("tinylist *2", tinylist * 2)
print("list + tinylist", list + tinylist)
list[1] = 100
print("list:", list)
list.append("added data")
print("list:", list)
```

list使用方括號

串列的第一個元素

索引2後的所有元素

兩個串列合併

附加在最後面

Run: main

```
list: ['Thomson', 78, 12.28, 'Sunny', 182.2]
list[0]: Thomson
list[1:3]: [78, 12.28]
list[2:]: [12.28, 'Sunny', 182.2]
tinylist *2 [123, 'Tony', 123, 'Tony']
list + tinylist ['Thomson', 78, 12.28, 'Sunny', 182.2, 123, 'Tony']
list: ['Thomson', 100, 12.28, 'Sunny', 182.2]
list: ['Thomson', 100, 12.28, 'Sunny', 182.2, 'added data']
```

tuple

使用逗點分隔元素

```
tuple = ('Thomson', 78, 12.28, 'Sunny', 182.2)
tinytuple = (123, 'Tony')
print("tuple:", tuple)
print("tuple[0]:", tuple[0])
print("tuple[1:3]:", tuple[1:3])
print("tuple[2:]:", tuple[2:])
print("tinytuple *2", tinytuple * 2)
print("tuple + tinytuple", tuple + tinytuple)
# tuple[1] = 100
print("tuple:", tuple)
#tuple.append("added data")
print("tuple:", tuple)
```

tuple使用小括號

tuple不支援assignment

tuple不能附加單一元素

```
tuple: ('Thomson', 78, 12.28, 'Sunny', 182.2)
tuple[0]: Thomson
tuple[1:3]: (78, 12.28)
tuple[2:]: (12.28, 'Sunny', 182.2)
tinytuple *2 (123, 'Tony', 123, 'Tony')
tuple + tinytuple ('Thomson', 78, 12.28, 'Sunny', 182.2, 123, 'Tony')
tuple: ('Thomson', 78, 12.28, 'Sunny', 182.2)
tuple: ('Thomson', 78, 12.28, 'Sunny', 182.2)
```

dictionary

A set includes paired key and value

字典使用大括號

key : value

```
dict = {}
dict['one'] = "This is one"
dict[2] = "This is two"
tinydict = {'name': 'tony', 'age': 24, 'height': 177}
print("tinydict", tinydict)
print("tinydict.keys()",tinydict.keys())
print("tinydict.values()",tinydict.values())
print("dict.keys()",dict.keys())
print("dict['one']",dict['one'])
print("dict[2]",dict[2])
```

對應到'one'的value

對應到2的value

```
tinydict {'name': 'tony', 'age': 24, 'height': 177}
tinydict.keys() dict_keys(['name', 'age', 'height'])
tinydict.values() dict_values(['tony', 24, 177])
dict.keys() dict_keys(['one', 2])
dict['one'] This is one
dict[2] This is two
```

```
list_of_words = ["hello","yes","goodbye","last"]
print(list_of_words[0]+","+list_of_words[1]+","+list_of_words[2]+","+list_of_words[3])
```

```
hello,yes,goodbye,last

Process finished with exit code 0
```

for

```
list_of_words = ["hello","yes","goodbye","last"]
print(list_of_words[0]+","+list_of_words[1]+","+list_of_words[2]+","+list_of_words[3])

to_print = ""
for w in list_of_words:
    to_print += w
    if w != list_of_words[-1]:
        to_print += ","
print(to_print)
```

```
hello,yes,goodbye,last
hello,yes,goodbye,last
```

將w指定為串列中的每一個元素，照順序代入for迴圈

附加w

list_of_words[-1]代表串列中的最後一個元素
檢查w是否不等於最後一個元素

join

```
list_of_words = ["hello","yes","goodbye","last"]
print(list_of_words[0]+","+list_of_words[1]+","+list_of_words[2]+","+list_of_words[3])

to_print = ""
for w in list_of_words:
    to_print += w
    if w != list_of_words[-1]:
        to_print += ","
print(to_print)

print(",".join(list_of_words))
```

將串列中的元素，合併成一個字串，元素與元素將以逗號分隔

```
hello,yes,goodbye,last
hello,yes,goodbye,last
hello,yes,goodbye,last
```

```python
list_of_words = ["Math","Calculus","Statistics","Algebra"]

ss = "-".join(list_of_words)
print(ss)

list_new = ss.split("-")
print(list_new)
```

將串列中的元素連結成一個字串，元素與元素間以 “-“ 分隔

以“-“為分隔符號，將字串分解為元素，儲存在串列中

```
Math-Calculus-Statistics-Algebra
['Math', 'Calculus', 'Statistics', 'Algebra']
```

sort

```
list_of_words = ["Math","Calculus","Statistics","Algebra"]

ss = "-".join(list_of_words)
print(ss)

list_new = ss.split("-")
print(list_new)

list_of_words.sort()
print(list_of_words)
```

將串列中的字串元素排序

```
Math-Calculus-Statistics-Algebra
['Math', 'Calculus', 'Statistics', 'Algebra']
['Algebra', 'Calculus', 'Math', 'Statistics']
```

reverse

```
list_of_words = ["Math","Calculus","Statistics","Algebra"]

ss = "-".join(list_of_words)
print(ss)

list_new = ss.split("-")
print(list_new)

list_of_words.sort()
print(list_of_words)

list_of_words.reverse()
print(list_of_words)
```

反轉串列中的元素順序

```
Math-Calculus-Statistics-Algebra
['Math', 'Calculus', 'Statistics', 'Algebra']
['Algebra', 'Calculus', 'Math', 'Statistics']
['Statistics', 'Math', 'Calculus', 'Algebra']
```

關鍵字 __name__

modu_1.py

```
print("this is modu_1")
if __name__ == '__main__':
    print("it is a main module")
else:
    print("it is an imported module")
```

Execute

__name__ 儲存
的內容是 __main__

```
this is modu_1
it is a main module
```

__name__

modu_1.py

```
print("this is modu_1")
if __name__ == '__main__':
    print("it is a main module")
else:
    print("it is an imported module")
```

__name__ 儲存
的內容不是 __main__

modu_2.py

```
print("this is modu_2")
if __name__ == '__main__':
    print("it is a main module")
else:
    print("it is an imported module")

import modu_1
```

Execute

__name__ 儲存
的內容是 __main__

```
this is modu_2
it is a main module
this is modu_1
it is an imported module
```

Get an odd number

```
while True:
    ss = input("please input an odd number (between 0-100):")
    try:
        num = int(ss)
    except:
        print("Invalid number, please input again.")
        continue
    if num%2 == 1:
        print(ss+" is odd")
        break
    else:
        print(ss+" is even. Input again")
```

輸入字串

將字串轉為整數

except代表轉換失敗

continue繼續到while迴圈的進入指令執行

執行直到迴圈進入條件不成立

如果餘數是1，代表奇數，break會中斷迴圈

```
please input an odd number (between 0-100):we2
Invalid number, please input again.
please input an odd number (between 0-100):22
22 is even. Input again
please input an odd number (between 0-100):21
21 is odd
```


while True:
continue
break
continue繼續到while
迴圈的進入指令執行
break停止執行while
迴圈

```
while True:
    try:
        num = int(ss)
    except:
        print("Invalid number, please input again.")
        continue
```

將輸入字串轉換為整數，可能出現例外狀況，就是使用者輸入字串中含有非數字的文字

當例外狀況發生時，列印訊息，繼續執行迴圈

```
while True:
    ss = input("please input an odd number (between 0-100):")
    try:
        num = int(ss)
    except:
        print("Invalid number, please input again.")
        continue
    if num%2 == 1:
        print(ss+" is odd")
        break
    else:
        print(ss+" is even. Input again")
```

轉換成功後，判斷是否為奇數，如果是奇數，break會停迴圈止執行

否則，列印訊息，並自動繼續執行迴圈