



深度學習TensorFlow實務

ResNet 圖片辨識

Lab3

-TA-

李偉弘

廖宜健

林佑昌

蔡明諺

彭冠偉

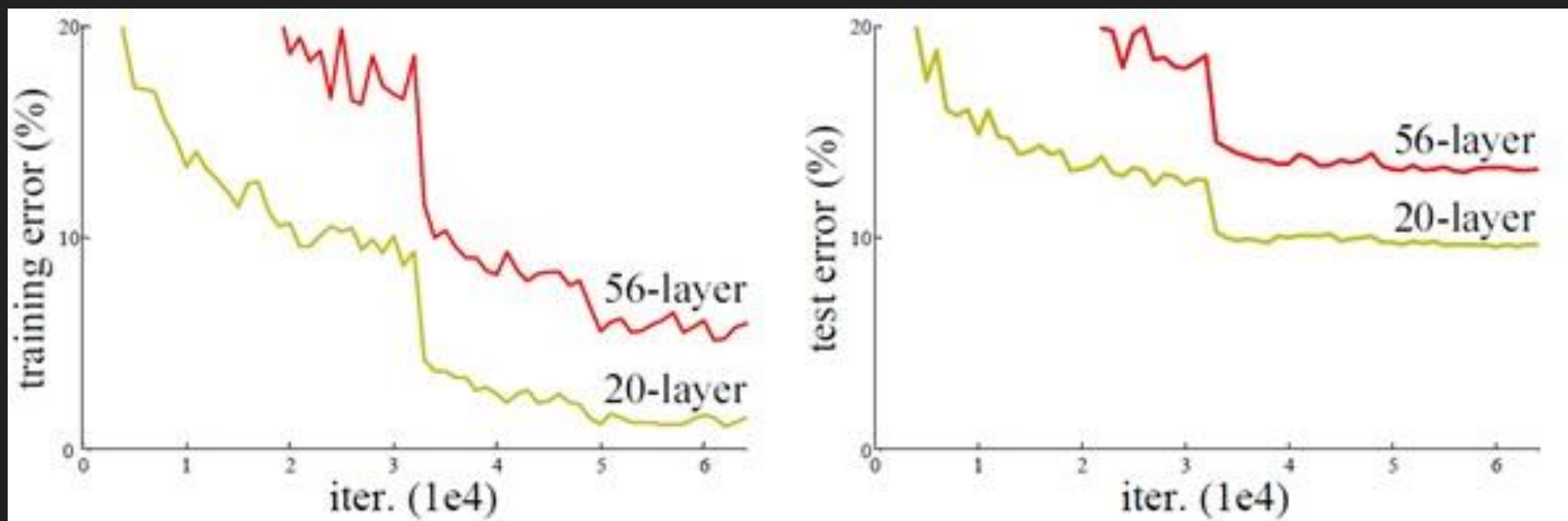
1. 應用情景

應用情景

- 對於深度學習來說，網路越深所能學到的東西越多，收斂速度變慢，訓練時間也越長。然而深度到達一定程度之後就會發現會有網路層越深，學習率越低、準確率無法提升的問題
- 這樣的問題我們稱為網路退化問題，甚至在某些情況下，網路層數的增加，反而會降低正確率

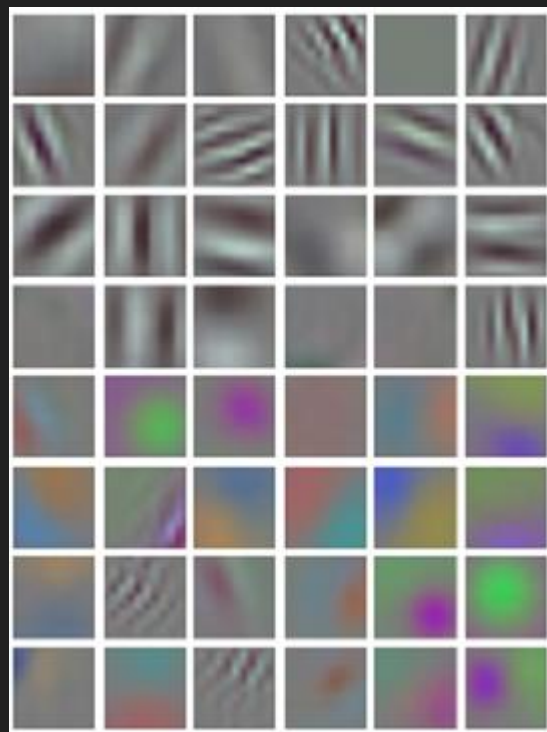
應用情景

- 在CIFA-10資料集上使用56層的卷積網路，其錯誤率無論是訓練集上還是測試集上，都遠高於20層的卷積神經網路



應用情景

- 圖片再經過多層卷積的採樣後，會出現一些現象，明明就是不同的圖片類別，但卻產生了看起來比較相似的特徵
- 這種差距的減少，也就使的最後得到的分類效果不理想



應用情景

- 那要怎麼解決網路退化問題？
- 解決辦法：在卷積神經網路中，引入可以刺激差異性和解決廣義化能力-深度殘差網路(ResNet)
- 到目前為止，在圖像分類(image classification)、物件偵測(object detection)、語意分割(semantic segmentation)等應用領域中，Resnet都表現出了良好的效果

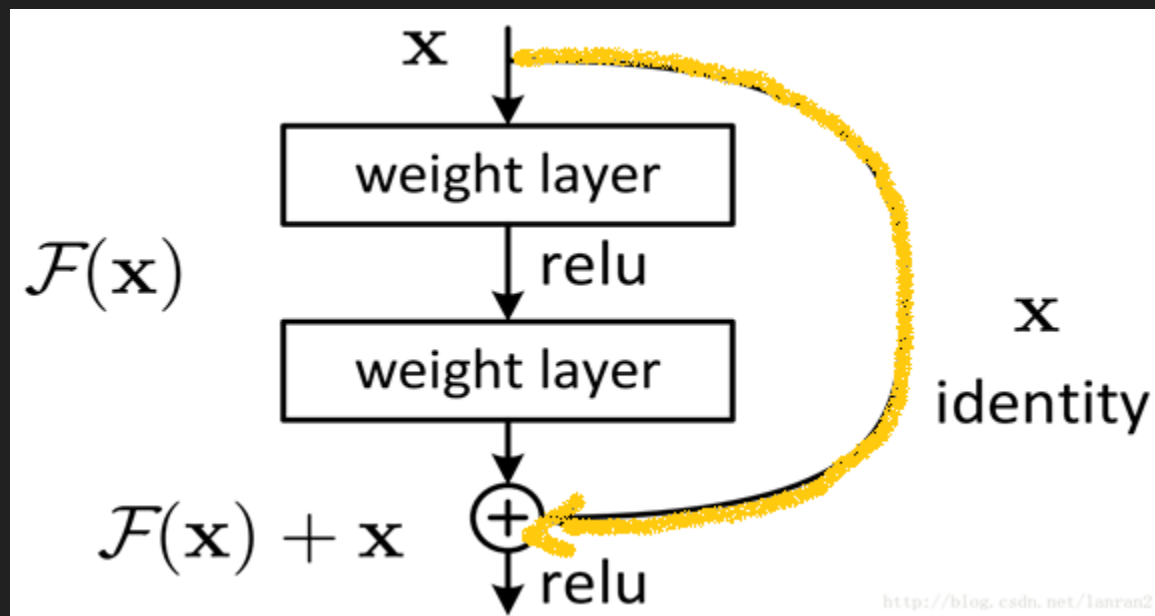
2. 結構解釋與數學推導

結構解釋

- 隨著網路深度加深，很容易出現梯度消失和梯度爆炸的問題
- 前面CNN有提到，在每一層的輸入透過卷積核後，都會產生類似於，有損壓縮的效果，壓縮到一定程度後，會分不清楚兩張照片，並不是甚麼意外的事情
- 這種行為叫有損壓縮並不適合，實際在工程上我們稱為降取樣(downsampling)，就是在向量跑過網路模型的過程中，經過一些濾波器(filter)處理，產生的效果就是讓輸入向量，透過降取樣處理後具有更小的尺寸，在CNN中常見的就是卷積層和池化層

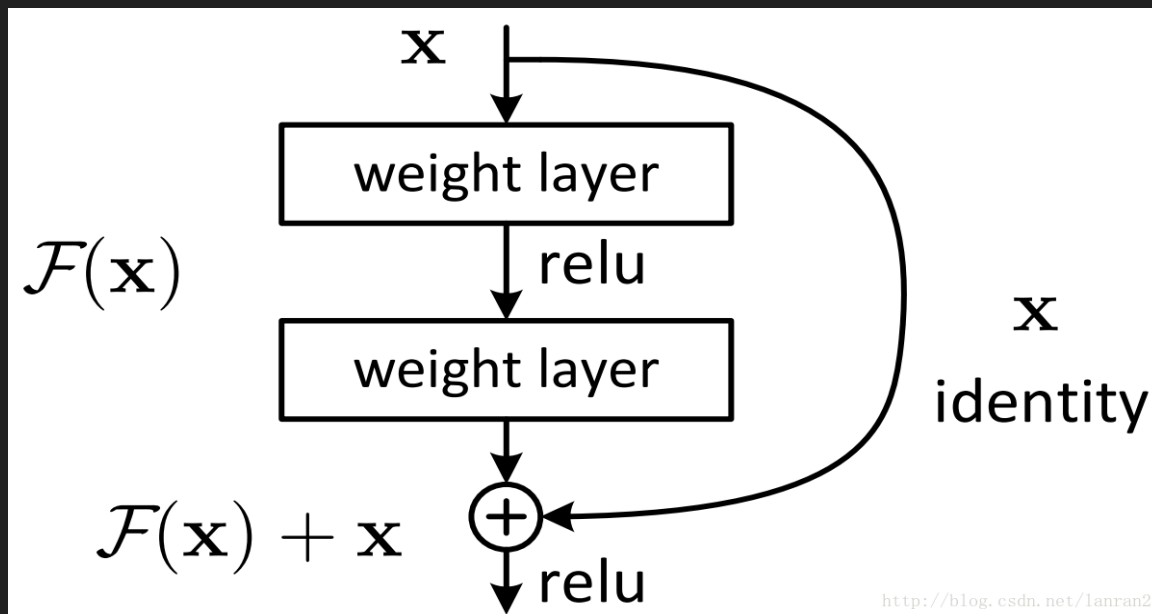
結構解釋

- 降取樣主要避免overfitting，以及減少一定的運算量
- ResNet中降取樣的方式，結構上出現了比較明顯的變化，它使用了類似「短路」式的設計，它將前面輸出的資料，直接跳過多層而引入到後面資料層輸入的部分



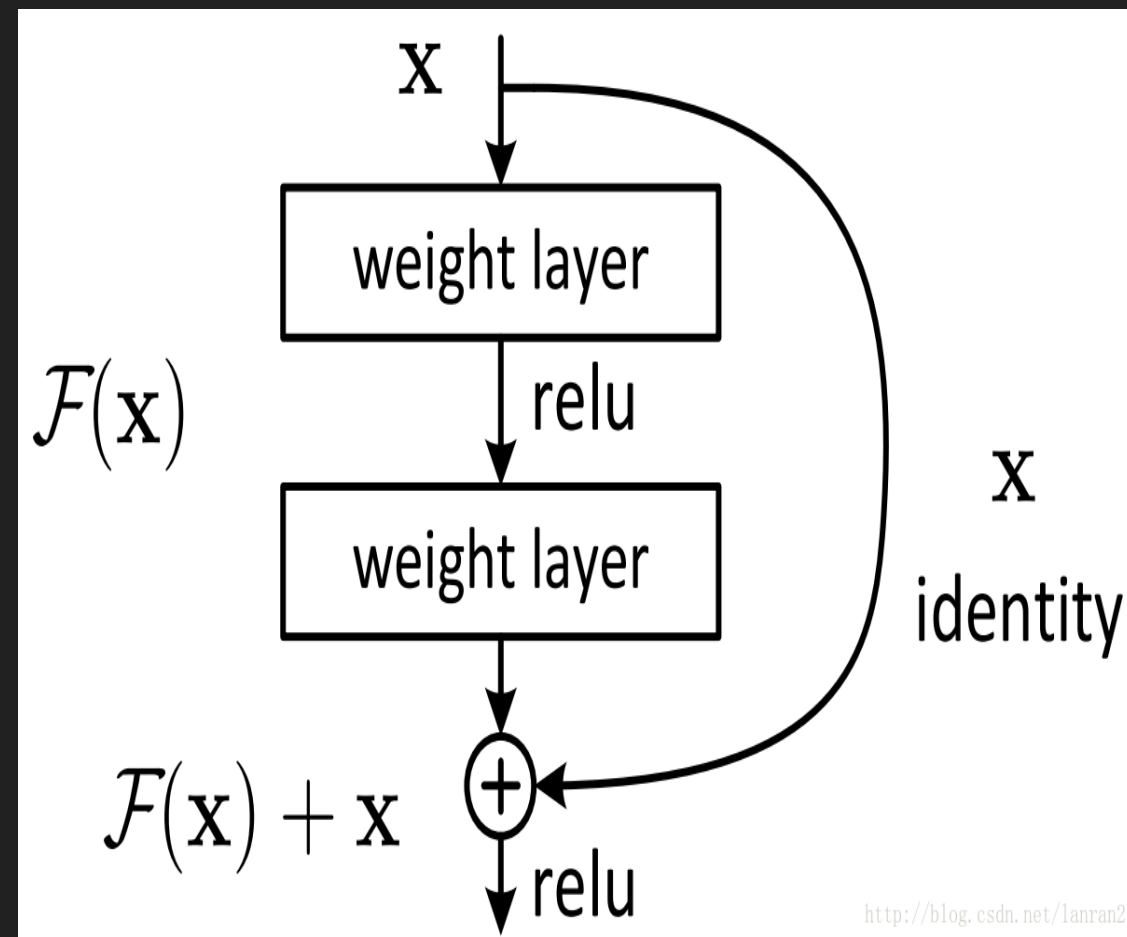
結構解釋

- 這樣做有個好處，前面層較為「清晰」的向量資料，會和後面被進一步「降取樣」過的資料，共同做為後面的資料輸入
- 假設由2層網路組成的關係，我們用 $F(x)$ 來代表，現在我們期望用 $H(x) = F(x) + x$ 來進行下一層的輸入，這樣的一個行為引入了更為豐富的參考訊息或著說更豐富的維度(特徵值)，這樣網路更能學到更為豐富的內容



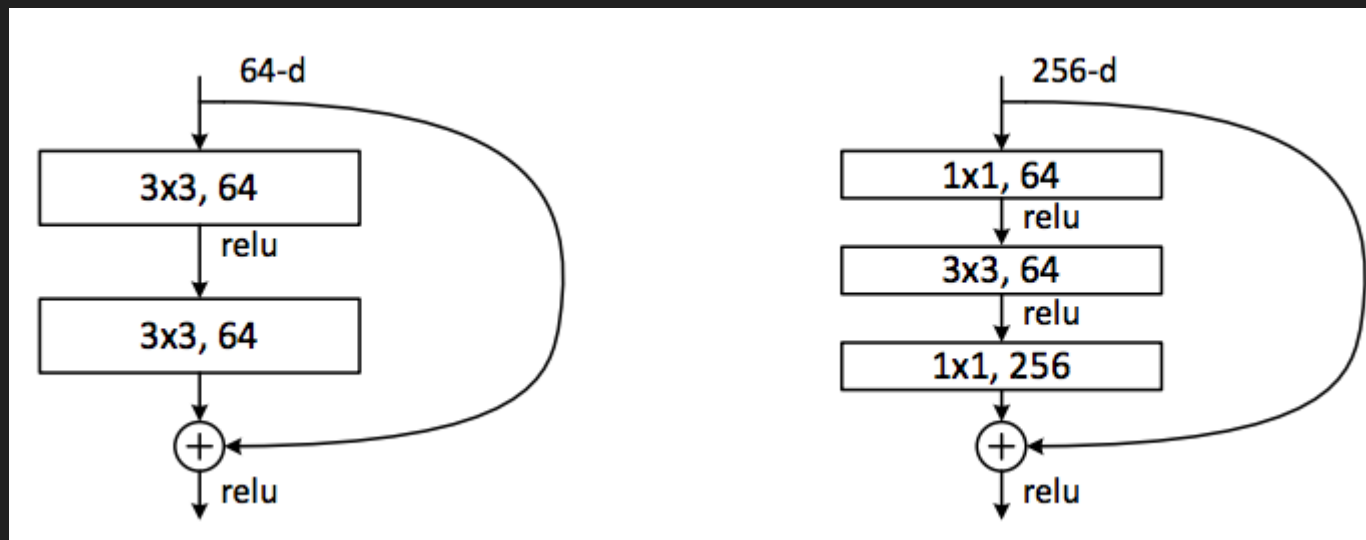
結構解釋

- ResNet分為兩部分，一部分為identity，另一部分為residual
- identity 指的是 x 部分
- residual 指的是 “剩餘” 也就是 $F(x)$
- ResNet只需要學習輸入、輸出差別的那一部分
- 殘差指的就是 $F(x)$ 部分



結構解釋

- Resnet除了兩層的殘差學習單元，還有三層的殘差學習單元，分別針對RestNet34(左圖) 和 RestNet50/101/152(右圖)
- 左圖兩層的殘差網路，包含兩個相同輸出通道數的64個3X3卷積核(因為殘差等於目標輸出減去輸入，即 $F(x) = H(x) - x$)

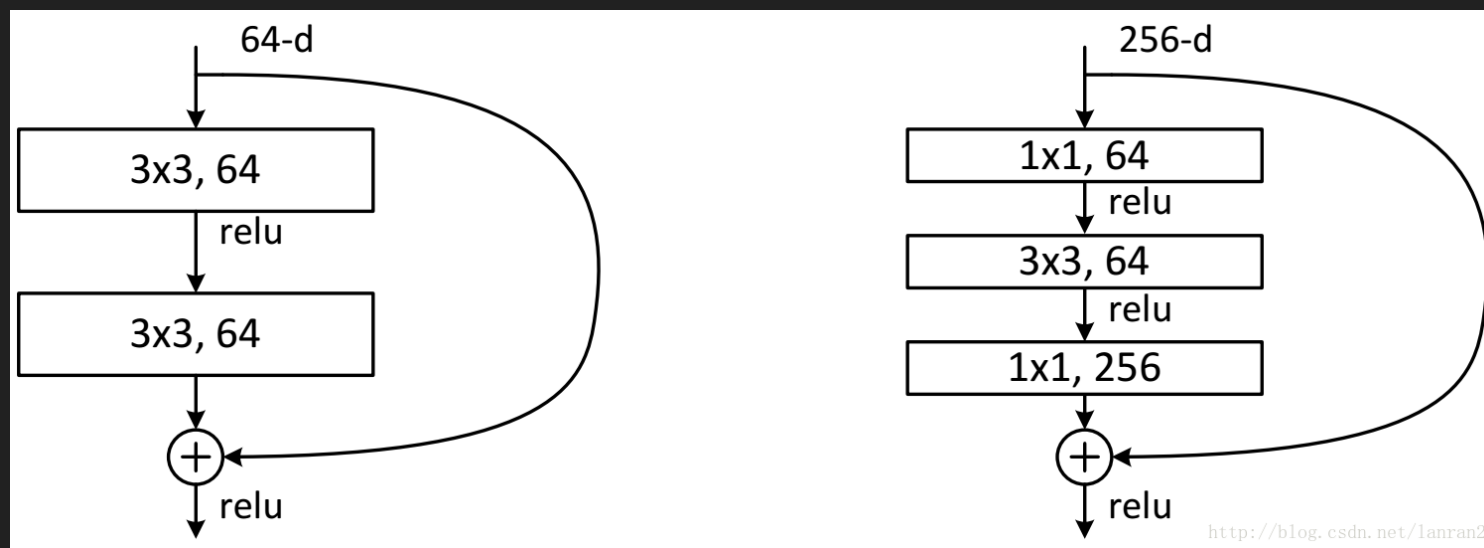


結構解釋

$$3 \times 3 \times 256 (\text{卷積核}) \times 256 (\text{輸入}) + 3 \times 3 \times 256 \times 256 = 1179648$$

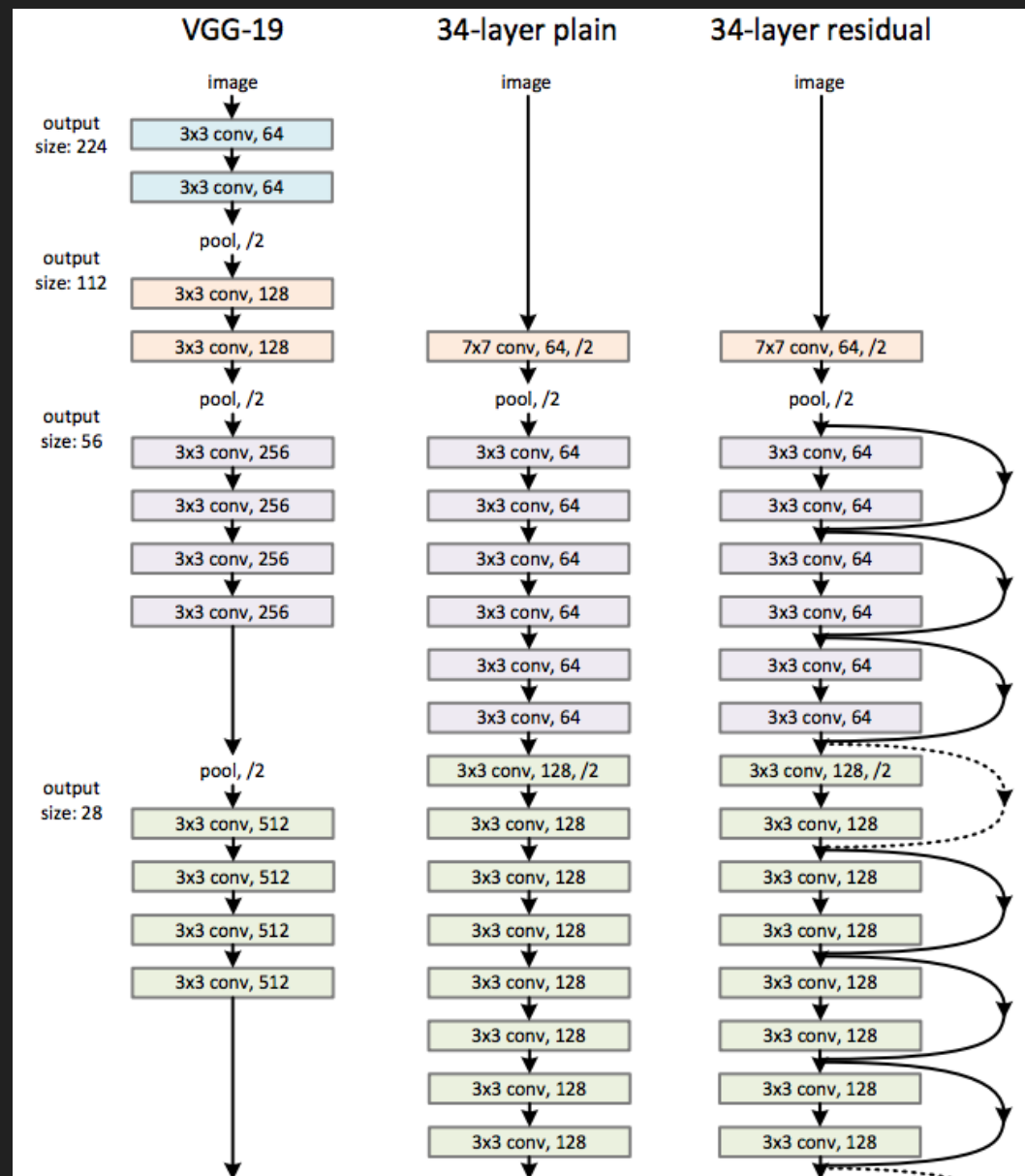
- 右圖又被稱為瓶頸設計(bottleneck design)，可以降低參數的數量，第一個1x1的卷積把256維channel降到64維，最後再通過1x1卷積恢復

$$1 \times 1 \times 64 \times 256 + 3 \times 3 \times 64 \times 64 + 1 \times 1 \times 256 \times 64 = 69632$$



結構解釋

- 右圖分別比較三種網路的深度和結構特點分別為 VGG19、34層的CNN普通網路、34層的深度殘差網路
- ResNet單純加深網路，所有的卷積層採用3x3的卷積核，不會在隱層設計任何的全連接層



數學推導

$$H(x) = F(x) + x$$

- ResNet有一個有趣的現象，我們可以觀察到 X_{l+1} 和前面一層 X_l ，純粹是一個線性疊加關係

$$X_{l+1} = F(X_l) + X_l$$



$$X_{l+2} = F(X_{l+1}) + X_{l+1}$$

$$X_{l+2} = F(X_{l+1}) + F(X_l) + X_l$$



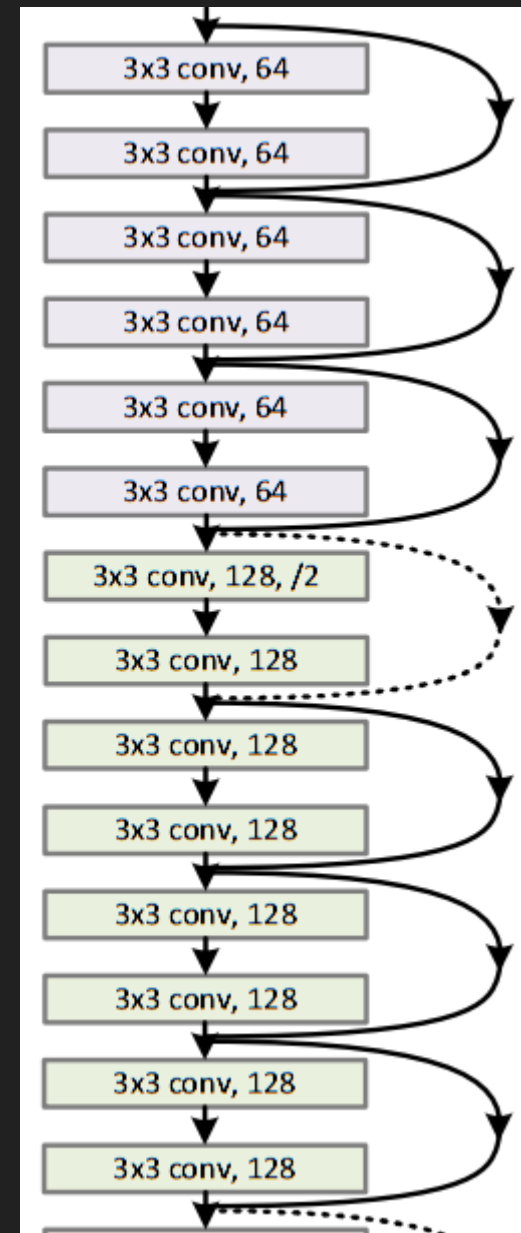
$$X_L = \sum_{i=1}^{L-1} F(X_i) + X_l$$

X_l

X_{l+1}

X_{l+2}

X_L



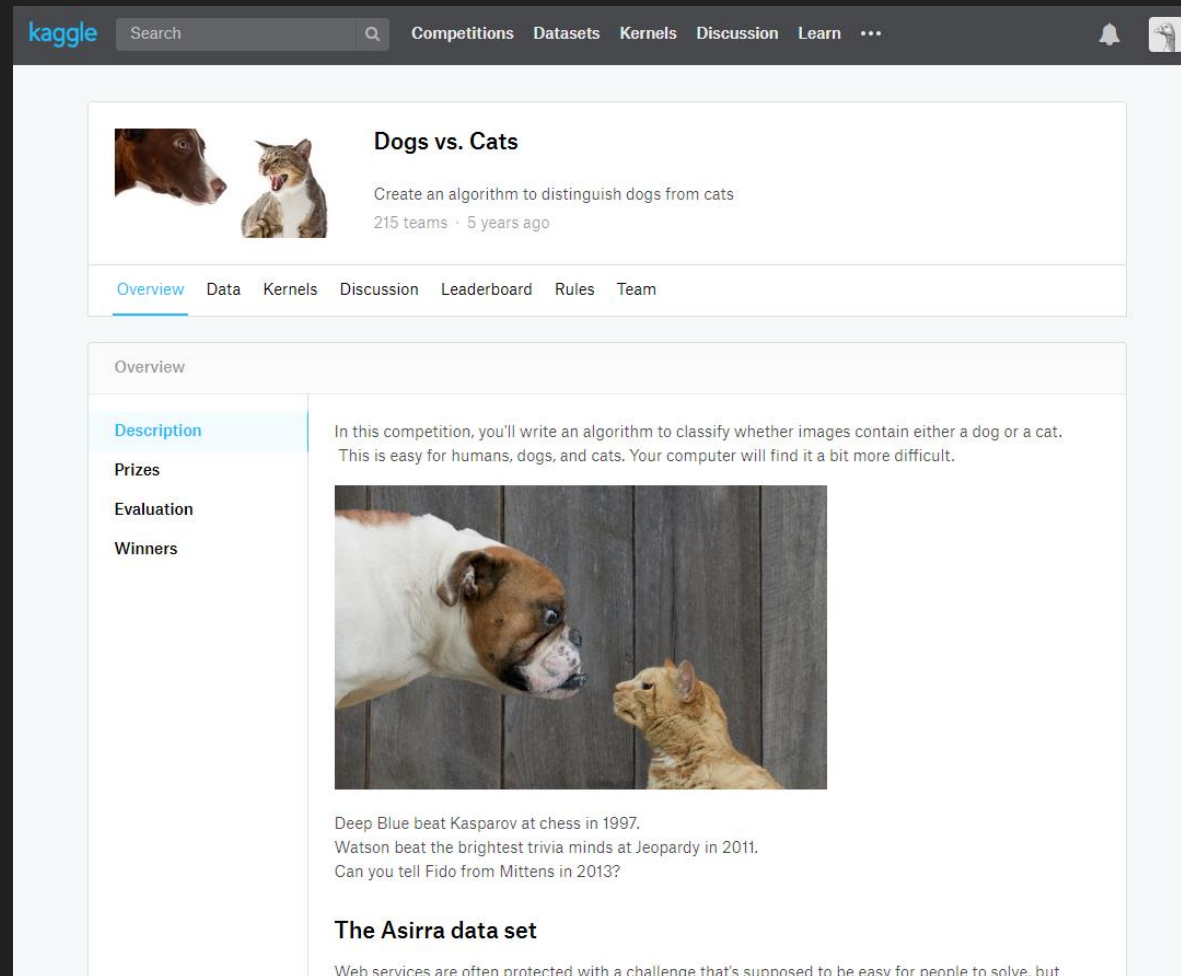
3. Cats v.s Dogs介紹

Cats v.s Dogs數據集

- Cats vs. Dogs是Kaggle大數據競賽某一年的一道賽題，利用給定的數據集(貓和狗圖片數量分別有12500張)，用演算法完成貓和狗的識別
- 這邊先幫同學下載部分資料集，並分成貓狗各500張訓練集、200張驗證集
- 兩個不同類別的標籤：
分別為 cat 和 dog

Cats v.s Dogs官方網站

■ <https://www.kaggle.com/c/dogs-vs-cats>

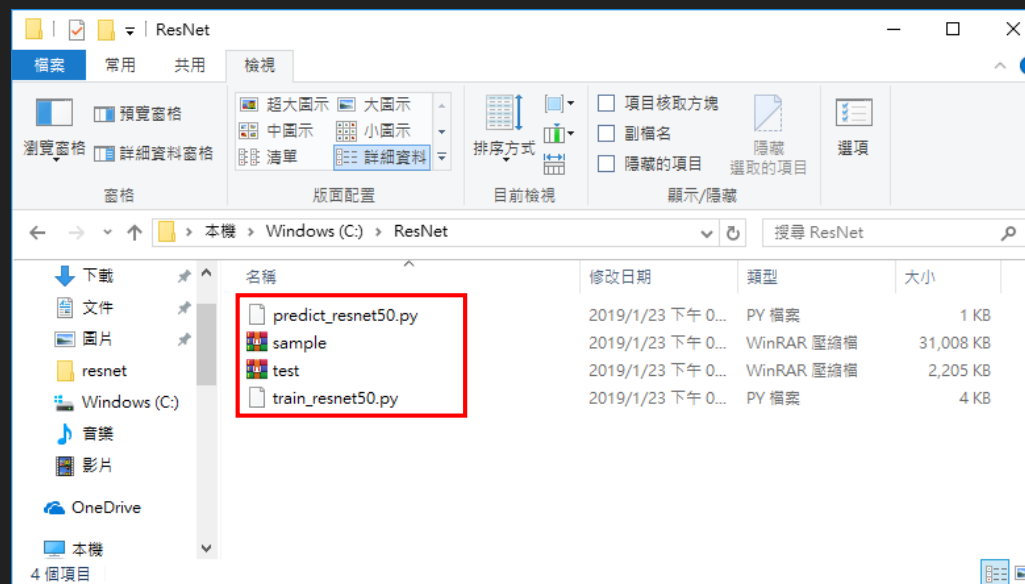
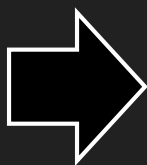
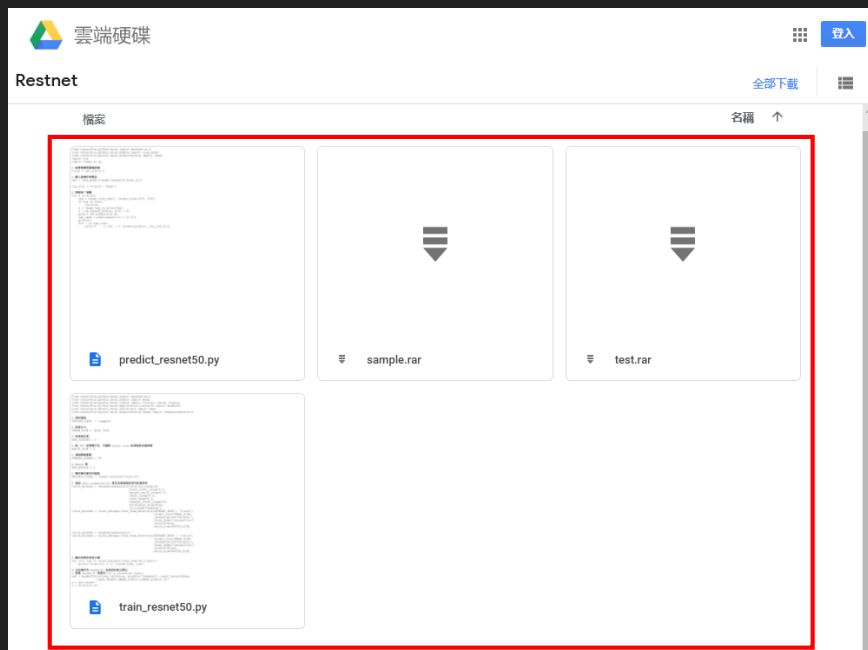


The screenshot shows the Kaggle website interface for the 'Dogs vs. Cats' competition. At the top, the Kaggle logo and navigation links (Search, Competitions, Datasets, Kernels, Discussion, Learn) are visible. The competition title 'Dogs vs. Cats' is prominently displayed, accompanied by a small image of a dog and a cat. Below the title, a description states: 'Create an algorithm to distinguish dogs from cats'. It also mentions '215 teams · 5 years ago'. A horizontal menu bar includes 'Overview', 'Data', 'Kernels', 'Discussion', 'Leaderboard', 'Rules', and 'Team'. The 'Overview' section is active, showing a sidebar with links to 'Description', 'Prizes', 'Evaluation', and 'Winners'. The main content area under 'Description' explains the competition goal: 'In this competition, you'll write an algorithm to classify whether images contain either a dog or a cat. This is easy for humans, dogs, and cats. Your computer will find it a bit more difficult.' Below this text is a large image of a bulldog and a ginger cat facing each other. Further down, a paragraph mentions 'Deep Blue beat Kasparov at chess in 1997. Watson beat the brightest trivia minds at Jeopardy in 2011. Can you tell Fido from Mittens in 2013?'. The section 'The Asirra data set' is partially visible at the bottom, with the text 'Web services are often protected with a challenge that's supposed to be easy for people to solve, but'.

4. 牛刀小試-ResNet做圖片分類

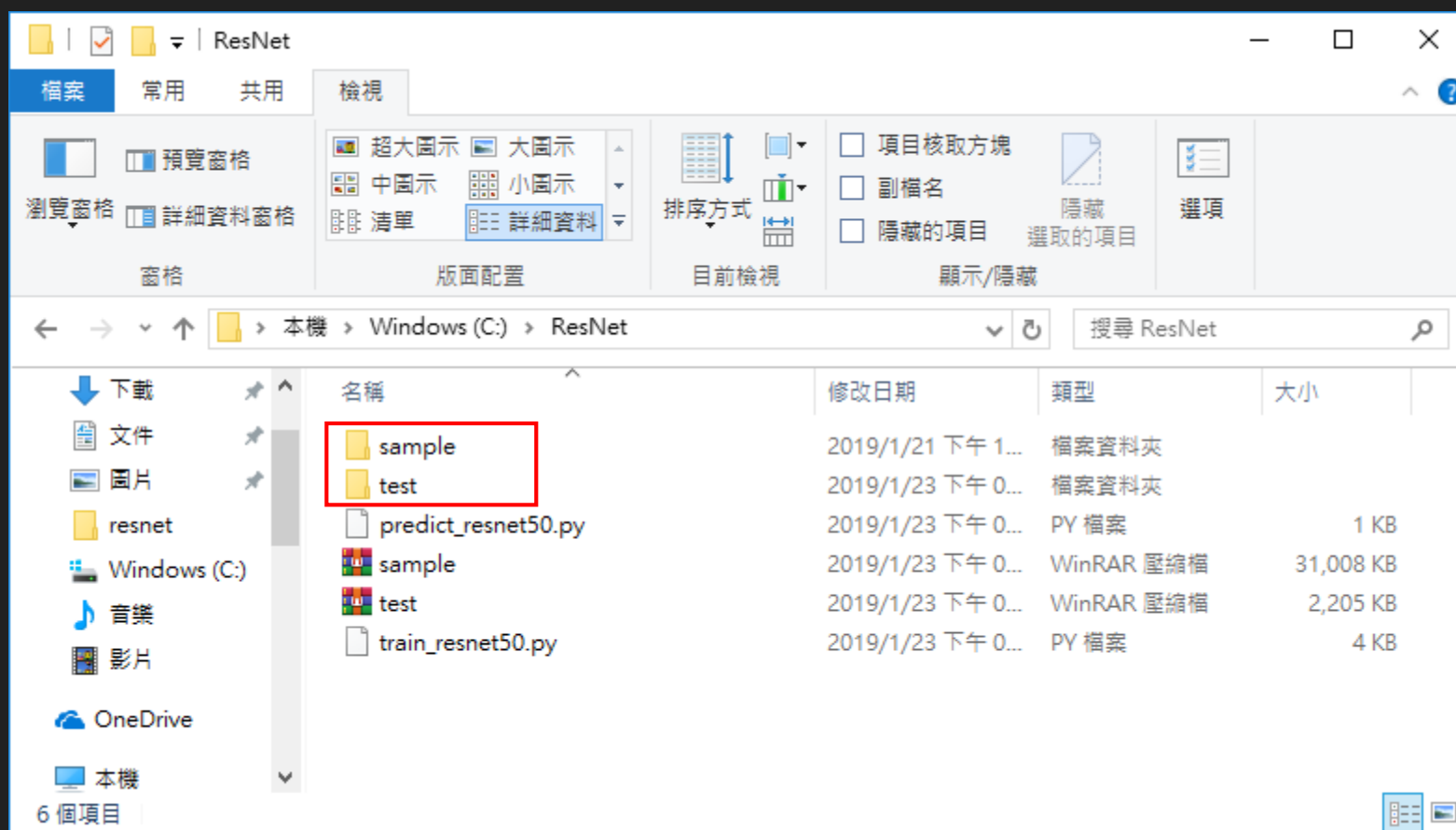
牛刀小試-ResNet做圖片分類

- 到https://drive.google.com/drive/folders/1rL5A-aToQhIUdun_qS_2EHZXxvEp1rgJ下載 train_resnet50.py 、 predict_resnet50.py 、 sample.rar 、 test.rar至 C:\ ResNet目錄之下



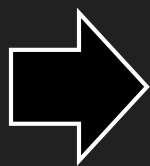
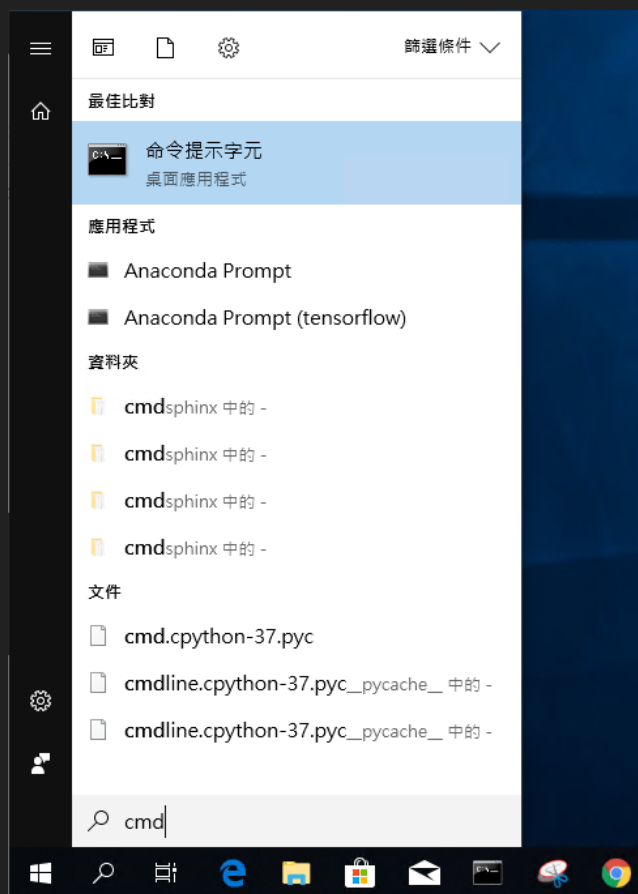
牛刀小試-ResNet做圖片分類

- 將sample.rar、test.rar “分別” 右鍵 “解壓縮至此”



牛刀小試-ResNet做圖片分類

- 在搜尋輸入 cmd，開啟命令提示字元




牛刀小試-ResNet做圖片分類

- 在 cmd 視窗，輸入以下指令

> cd C:\ResNet

- 工作路徑由 C:\Users\NTUT 變成 C:\ResNet

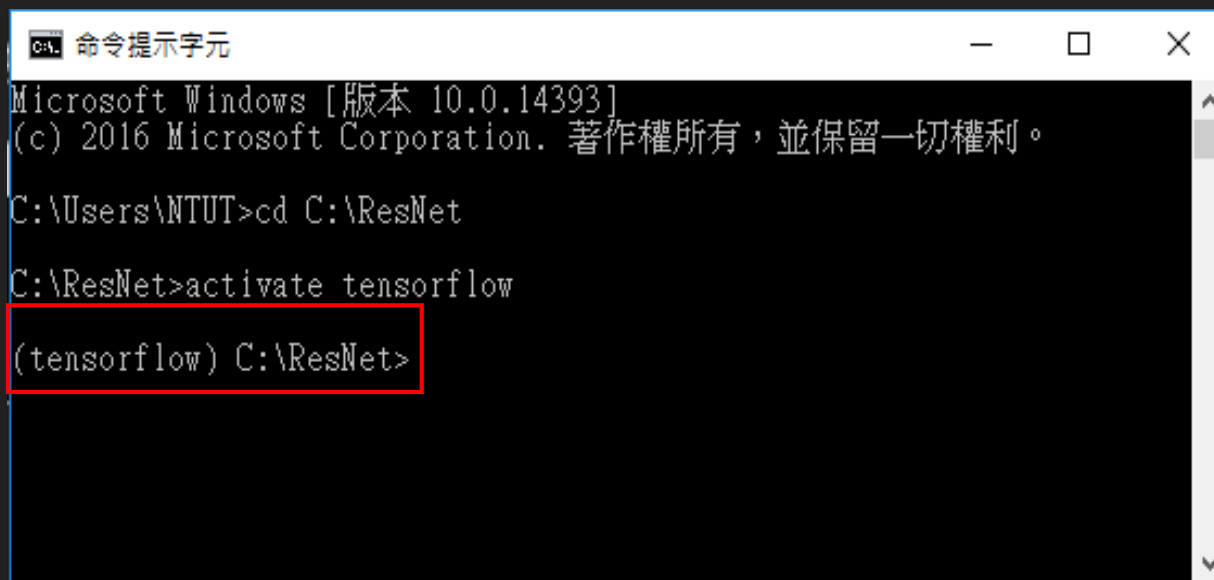


```
命令提示字元
Microsoft Windows [版本 10.0.14393]
(c) 2016 Microsoft Corporation. 著作權所有，並保留一切權利。
C:\Users\NTUT>cd C:\ResNet
C:\ResNet>
```

牛刀小試-ResNet做圖片分類

- 在 cmd 視窗，輸入以下指令

> **activate tensorflow** # 之前已經建立該虛擬環境



```
命令提示字元
Microsoft Windows [版本 10.0.14393]
(c) 2016 Microsoft Corporation. 著作權所有，並保留一切權利。
C:\Users\NTUT>cd C:\ResNet
C:\ResNet>activate tensorflow
(tensorflow) C:\ResNet>
```


牛刀小試-ResNet做圖片分類

- 在 cmd 視窗，輸入以下指令

> python train_resnet50.py

```
命令提示字元 - python train_resnet50.py
Microsoft Windows [版本 10.0.14393]
(c) 2016 Microsoft Corporation. 著作權所有，並保留一切權利。

C:\Users\NTUT>cd C:\ResNet

C:\ResNet>activate tensorflow

(tensorflow) C:\ResNet>python train_resnet50.py
C:\Users\NTUT\Anaconda3\envs\tensorflow\lib\site-packages\h5py\__init__.py:36: FutureWarning: Conversion
nt of issubdtype from `float` to `np.floating` is deprecated. In future, it will be treated as `np.float
.type`.
  from ._conv import register_converters as _register_converters
Found 1000 images belonging to 2 classes.
Found 400 images belonging to 2 classes.
Class #0 = cats
Class #1 = dogs
2019-01-24 00:28:39.577955: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instr
sorFlow binary was not compiled to use: AVX AVX2

Layer (type)                 Output Shape          Param #    Connected to
-----
input_1 (InputLayer)         (None, 224, 224, 3)   0
conv1 (Conv2D)               (None, 112, 112, 64) 9472      input_1[0][0]
```

牛刀小試-Resnet做圖片分類

- 出現以下圖式代表訓練成功

```
命令提示字元
activation_46 (Activation)      (None, 7, 7, 512)    0      bn5c_branch2a[0][0]
res5c_branch2b (Conv2D)        (None, 7, 7, 512)    2359808 activation_46[0][0]
bn5c_branch2b (BatchNormalizati (None, 7, 7, 512)    2048    res5c_branch2b[0][0]
activation_47 (Activation)      (None, 7, 7, 512)    0      bn5c_branch2b[0][0]
res5c_branch2c (Conv2D)        (None, 7, 7, 2048)   1050624 activation_47[0][0]
bn5c_branch2c (BatchNormalizati (None, 7, 7, 2048)   8192    res5c_branch2c[0][0]
add_15 (Add)                   (None, 7, 7, 2048)   0      bn5c_branch2c[0][0]
                                activation_45[0][0]
activation_48 (Activation)      (None, 7, 7, 2048)   0      add_15[0][0]
avg_pool (AveragePooling2D)     (None, 1, 1, 2048)   0      activation_48[0][0]
flatten (Flatten)              (None, 2048)         0      avg_pool[0][0]
dropout (Dropout)              (None, 2048)         0      flatten[0][0]
softmax (Dense)                (None, 2)            4098    dropout[0][0]
Total params: 23,591,810
Trainable params: 22,933,250
Non-trainable params: 658,560
None
Epoch 1/1
500/500 [=====] - 1249s 2s/step - loss: 0.5531 - acc: 0.7000 - val_loss: 0.1927 - val_acc: 0.9250
(tensorflow) C:\ResNet>
微軟注音 半 :
```

牛刀小試-Resnet做圖片分類

- 接下來要對剛剛訓練的模型，進行準確率評估
- 在 cmd 視窗，輸入以下指令

```
> python predict_resnet50.py C:\ResNet\test\1.jpg
```

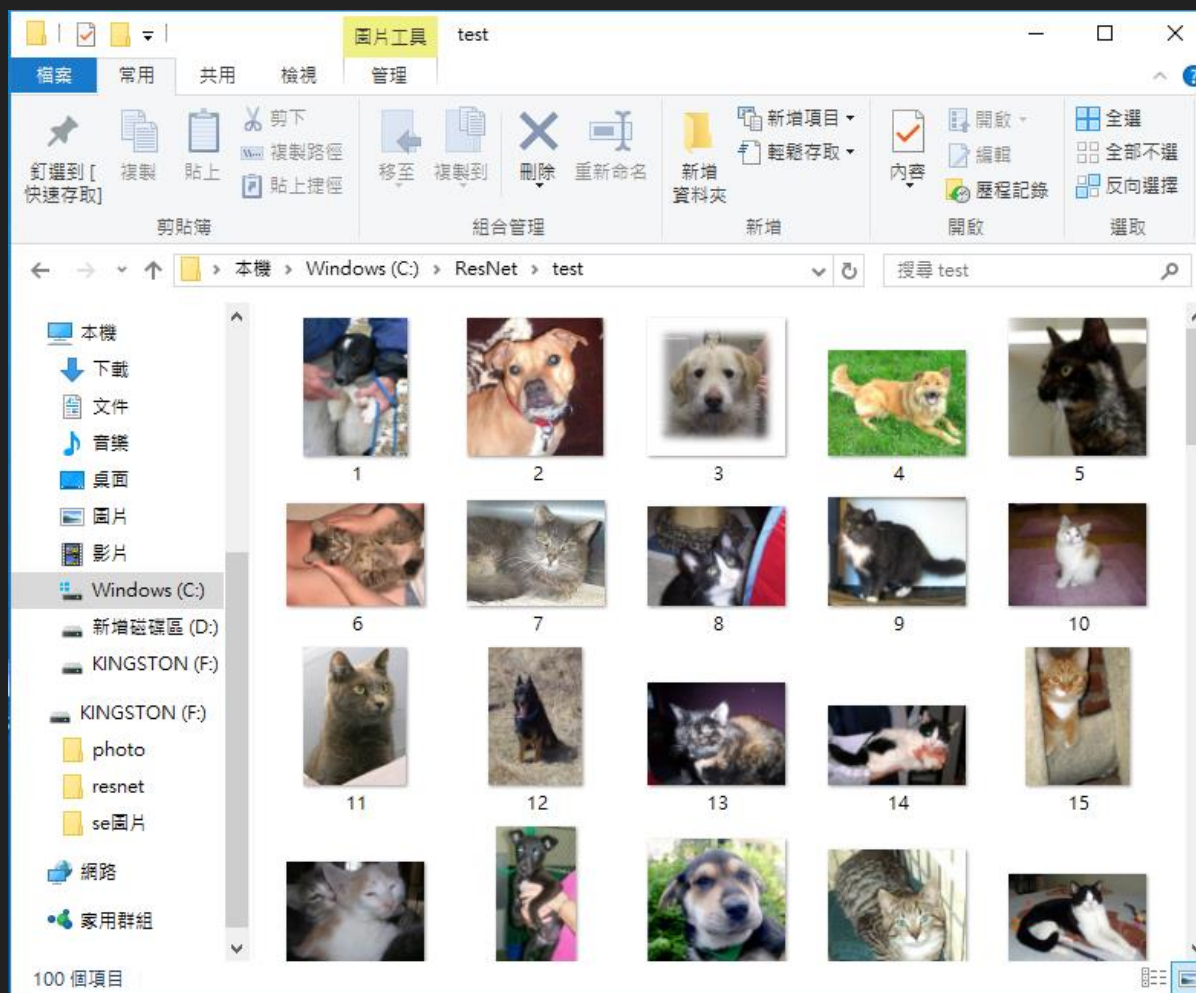
- 準確率 precision = 0.98



```
命令提示字元
(tensorflow) C:\ResNet>python predict_resnet50.py C:\ResNet\test\1.jpg
C:\Users\NTUT\Anaconda3\envs\tensorflow\lib\site-packages\h5py\__init__.py:36: FutureWarning: Conversion of the second argument of issubdtype from `float` to `np.floating` is deprecated. In future, it will be treated as `np.float64 == np.dtype(float).type`.
  from ._conv import register_converters as _register_converters
2019-01-24 00:52:30.393858: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX AVX2
C:\ResNet\test\1.jpg
  0.980  dogs
  0.020  cats
(tensorflow) C:\ResNet>
```

牛刀小試-Resnet做圖片分類

- 測試test資料夾裡的其他圖片，其結果為何？



5. train_resnet50.py 程式碼解析

train_resnet50.py 程式碼解析

```

DATASET_PATH = 'sample'
IMAGE_SIZE = (224, 224)
NUM_CLASSES = 2
BATCH_SIZE = 2
FREEZE_LAYERS = 45
NUM_EPOCHS = 1
WEIGHTS_FINAL = 'model-resnet50-final.h5'

train_datagen = ImageDataGenerator(rotation_range=40,
                                   width_shift_range=0.2,
                                   height_shift_range=0.2,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   channel_shift_range=10,
                                   horizontal_flip=True,
                                   fill_mode='nearest')

```

資料路徑

影像大小

影像類別數

模型輸出的檔案

透過資料增強產生訓練與驗證用的資料影像

cifar10_train.py 程式碼解析

```
for cls, idx in train_batches.class_indices.items():  
    print('Class #{} = {}'.format(idx, cls))
```

輸出各類別的索引值

```
net = ResNet50(include_top=False, weights='imagenet', input_tensor=None,  
               input_shape=(IMAGE_SIZE[0], IMAGE_SIZE[1], 3))  
x = net.output  
x = Flatten()(x)
```

以訓練好的ResNet50為基礎
來建立模型，捨棄ResNet50
頂層的全連接層

```
x = Dropout(0.5)(x)
```

增加DropOut Layer

```
output_layer = Dense(NUM_CLASSES, activation='softmax', name='softmax')(x)
```

```
net_final = Model(inputs=net.input, outputs=output_layer)  
for layer in net_final.layers[:FREEZE_LAYERS]:  
    layer.trainable = False  
for layer in net_final.layers[FREEZE_LAYERS:]:  
    layer.trainable = True
```

設定凍結與要進行訓練的網路層

cifar10_train.py 程式碼解析

```
net_final = Model(inputs=net.input, outputs=output_layer)
for layer in net_final.layers[:FREEZE_LAYERS]:
    layer.trainable = False
for layer in net_final.layers[FREEZE_LAYERS:]:
    layer.trainable = True
```

```
net_final.compile(optimizer=Adam(lr=1e-5),
                  loss='categorical_crossentropy', metrics=['accuracy'])
```

使用 Adam optimizer，以較低的 learning rate 進行 fine-tuning

```
print(net_final.summary())
```

輸出整個網路架構

```
net_final.fit_generator(train_batches,
                        steps_per_epoch = train_batches.samples // BATCH_SIZE,
                        validation_data = valid_batches,
                        validation_steps = valid_batches.samples // BATCH_SIZE,
                        epochs = NUM_EPOCHS)
```

訓練模型

```
net_final.save(WEIGHTS_FINAL)
```


8. predict_resnet50.py 程式碼解析

predict_resnet50.py 程式碼解析

```
files = sys.argv[1:]
```

從參數讀取檔案路徑

```
net = load_model('model-resnet50-final.h5')
```

載入訓練好的模型

```
cls_list = ['cats', 'dogs']
```

```
for f in files:
```

```
    img = image.load_img(f, target_size=(224, 224))
```

```
    if img is None:
```

```
        continue
```

```
    x = image.img_to_array(img)
```

```
    x = np.expand_dims(x, axis = 0)
```

```
    pred = net.predict(x)[0]
```

```
    top_inds = pred.argsort()[::-1][:5]
```

```
    print(f)
```

```
    for i in top_inds:
```

```
        print('    {:.3f} {}'.format(pred[i], cls_list[i]))
```

辨識每一張圖片

-END-