



卷積神經網路 Convolutional Neural Network & 電腦視覺 Computer Vision Part5

林彥宇&教研處

理論講授 [投影片](#)
今日課程 [投影片](#)
[影片播放列表](#)

「版權聲明頁」

本投影片已經獲得作者授權台灣人工智慧學校得以使用於教學用途，如需取得重製權以及公開傳輸權需要透過台灣人工智慧學校取得著作人同意；如果需要修改本投影片著作，則需要取得改作權；另外，如果有需要以光碟或紙本等實體的方式傳播，則需要取得人工智慧學校散佈權。

課程內容

本日課程：

- 1. Object Recognition and Detection**
- 2. R-CNN Family**
- 3. Faster R-CNN 手把手**

本次課程結束後你 (妳) 應該會什麼？

- **軟實力**

- 理解object detection的概念
- 了解R-CNN的方法與程式碼內容

- **硬底子**

- 完成手把手faster R-CNN
- 試著使用Label工具(optional)



Code / Data 放在 hub 中的 courses 內


- 為維護課程資料，courses 中的檔案皆為 read-only, 如需修改請 cp 至自身的環境中
- 打開 terminal, 輸入
 - [台北班]
`cp -r courses-tpe/CVCNN/part5 <存放至本機的名稱>`
 - [新竹班]
`cp -r courses-hsi/CVCNN/part5 <存放至本機的名稱>`
 - [台中班]
`cp -r courses-txg/CVCNN/part5 <存放至本機的名稱>`



Object Recognition and Detection

理論講授 - Fine-grained Object Recognition

Idea



Visualizing and understanding convolutional networks
[Matthew D., and Fergus, 2014]

- Neuron: part detector

Research Center for Information Technology Innovation, Academia Sinica



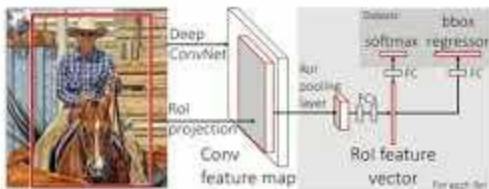
R-CNN Family

理論講授 - Object Detection

Fast R-CNN

[Girshick, ICCV'15]

• System overview



- Apply fully convolutional networks to the whole image
- RoI pooling: each proposal is pooled into a fix-size feature map
- Classification with a softmax layer
- Regression-based bounding box refinement

Research Center for Information Technology Innovation, Academia Sinica

111

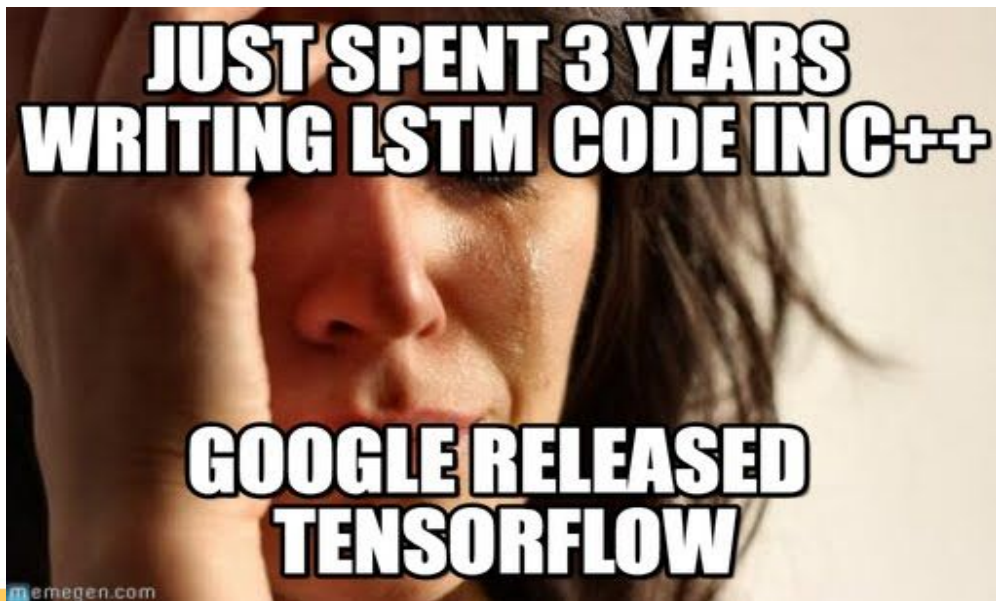


程式實作 - R-CNN in tensorflow



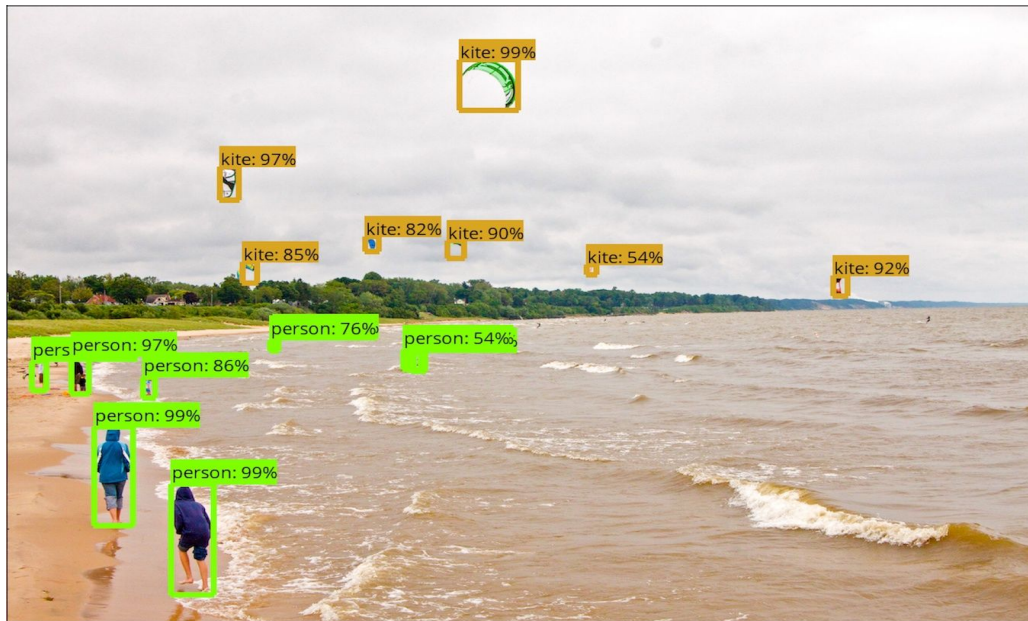
It's hard to implement two stage model...

- 裡面非常多細節以及相關的訓練技巧, paper 不一定會寫得很清楚
- 當然可以試著手刻以上的模型
 - [R-CNN](#)
 - [Fast R-CNN](#)
 - [Faster R-CNN](#)
 - [Mask R-CNN](#)



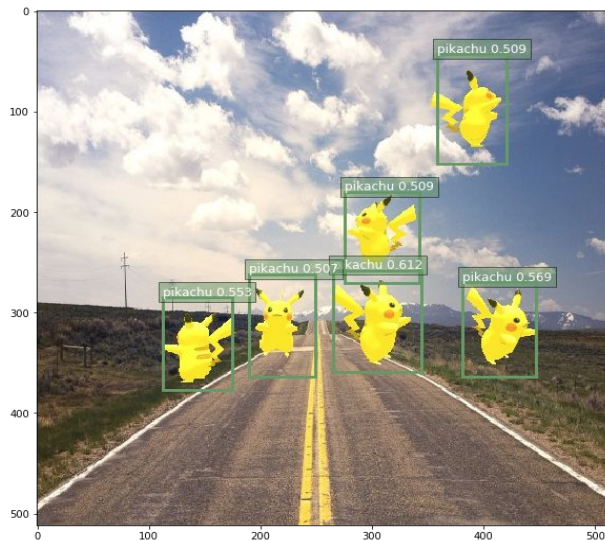
tensorflow object detection API

- 包含多數的主流模型，而且都已經訓練完畢
 - Faster R-CNN with ResNet 50
 - SSD with mobileNet
 - Mask R-CNN



How to inference

- 這些模型都已經在 COCO dataset 上訓練完畢，具備辨識了 **80 種 object 的能力** (勘誤: COCO 只有 80 個類別)
- 直接把我們要測試的圖片輸入模型，就能夠得到 detect 之後的結果



Object detection inference tutorial

- 請打開
CVCNN/part5/04_FasterRCNN/session1_2/fasterRcnn.ipynb,
試著了解裡面的程式碼
- 歡迎大家分享自己上傳圖片的 detect 結果



Object detection - training with other dataset

- 原本的模型是訓練在 COCO dataset 上面，因此只能夠辨識 COCO 資料裡面的 80 種物體 (勘誤: COCO 只有 80 個類別)
- 如果希望模型能夠辨識自己希望能辨別的物體，就必須提供新的資料集進行 fine-tune



物件偵測-手把手入門概要

(助教講解)

物件偵測-手把手入門概要

物件偵測

物件偵測-手把手入門概要(一)

Semantic Segmentation	Classification + Localization	Object Detection	Instance Segmentation
			

來自Stanford University CS231n
http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture11.pdf

圖像辨識除了一般的分類問題，還有上面的四種：

- Semantic segmentation：對各個pixel level 進行分類。
- Classification + Localization：除了分類結果並給予目標定位。
- Object detection：對於多目標進行分類並定位。
- Instance segmentation：對多目標分類後並進行pixel level分類。

按一下即可新增演講者備忘稿

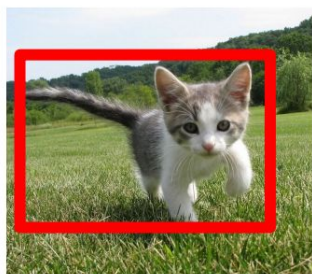


物件偵測-手把手入門概要(一)

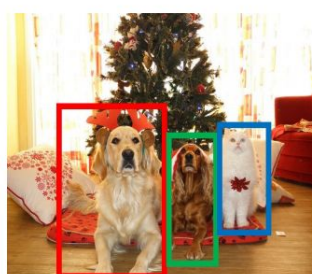
**Semantic
Segmentation**



**Classification
+ Localization**



**Object
Detection**



**Instance
Segmentation**



來自Stanford University CS231n

http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture11.pdf

圖像辨識除了一般圖像分類問題以外，還有上圖的四種：

Semantic segmentation：對各個像素區塊進行分類。

Classification + Localization：除了分類結果並給予目標定位。

Object detection：對於多目標進行分類並定位。

Instance segmentation：對多目標物件進行像素區塊分類。



物件偵測-手把手入門概要(二)

對於物件偵測來說，我們得到Ground truth大致上可以分成兩種類型，分別為**盒型(Boxes)**與**遮罩(Masks)**，**盒型**的輸出為物件類別在圖像中的長寬以及位置。

而**遮罩**如其名，除了能做到**盒型**能做到的部分，並且能夠讓該物件與背景準確的分出來。

除了輸出的差異以外，在使用不同作者的**開源碼**時，也要注意每個作者的**輸入**與Ground truth格式也都不盡相同。

Label軟體工具：

Boxes: [Labelimg](#)

Masks: [Labelme](#)



物件偵測-手把手入門概要(三)

這邊簡介物件偵測的模型架構的概念，目前主流上分為兩種，並不是指上一頁的箱型與遮罩。

主要分為One stage與Two stage，差異為，在做物件偵測時，定位框與類別是否一起進行推論。

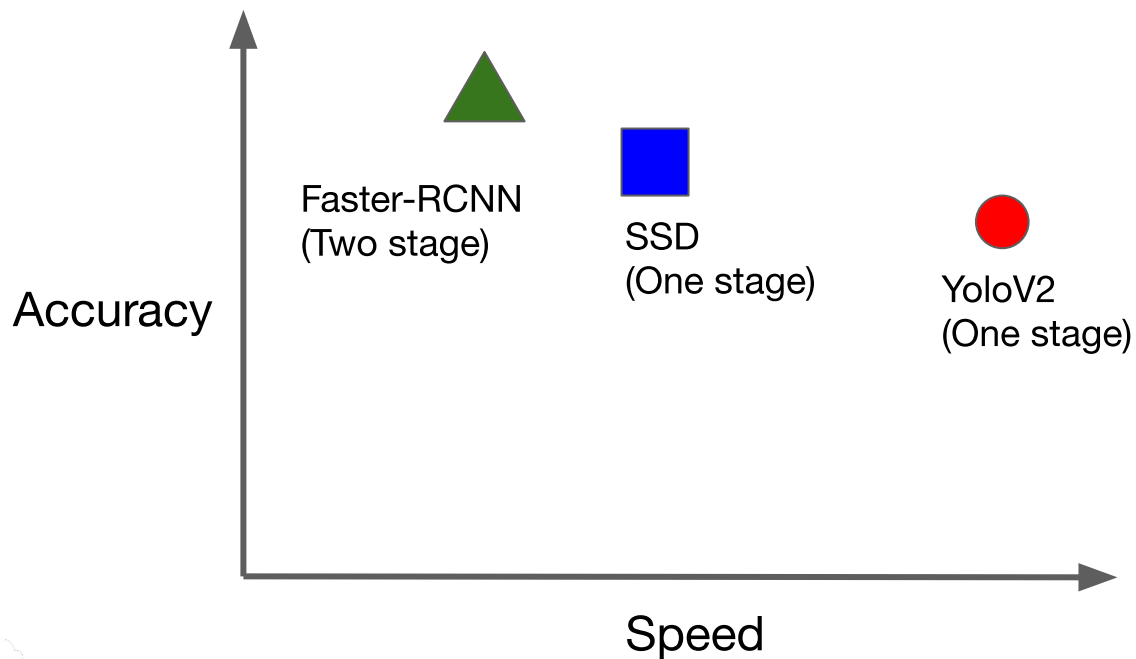
舉例來說，對於最先開始的 Two stage，是先找出物件的位置後，再將該物件往後面的分類器進行辨識，而One stage則是一次同時預測類別與定位框。

而理所當然的，two stag相對於one stage辨識率高，相對速度慢，反言之。

在下一頁有比較圖表。



物件偵測-手把手入門概要(四)



小天使提醒：
本章全部有下底線的
都附有方便的超連
結。

Faster-RCNN 手把手 快速上手

(助教講解)

此篇程式碼來源為以下網址，並進行修改

https://github.com/tensorflow/models/tree/master/research/object_detection

Tensorflow Object Detection API

在前面的課程有提到許多物件偵測的Model, Faster-RCNN的手把手教學中, 我們使用Tensorflow object detection API來實現, 在裡面, 提供了許多不同的物件偵測的腳本可以進行替換使用, 在之後的練習中, 學員們可以嘗試更換其他的Model試著使用看看。



Faster-RCNN手把手簡介(一)

Faster-RCNN是物件偵測的演算法的一種，輸出屬於**盒型(Boxes)**，由於是**Two Stage**的演算法，特點與YOLO相反，是屬於辨識準確度相對較高，速度相較為慢的演算法一種。



Faster-RCNN手把手簡介(二)

在本篇演算法中，我們以Tensorflow object detection API來作為範例，裡面有多數Box類型的物件偵測Model可以替換使用。

Tensorflow Object detection API 所使用的Label的Input格式為Record的特殊格式，必須先將PASCAL VOC先轉為Record才能輸入。

也由於API已經將多數運作腳本化，所以不在細節多加說明。

在Hub中有提供以下.ipynb，方便學員直接上手Faster-RCNN
1.fasterRcnn.ipynb



Faster-RCNN手把手簡介(三)

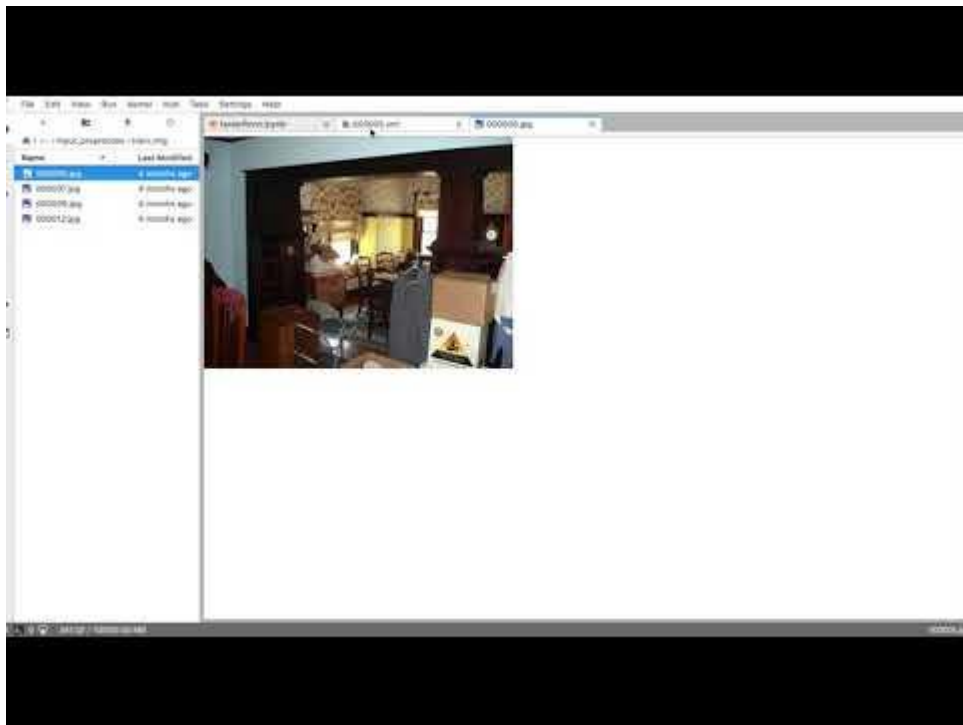
Tensorflow Object Detection API, 可以替換各種model, 如果想替換我們分別需要取得對應的[腳本](#)以及[Pretrain model](#)並且取代原本的腳本以及Pretrain model, 而Hub所提供的[腳本的程式碼內文](#)有因應Hub的路徑進行修改, 若要嘗試使用其他Model, 請依照原本的內容進行修改。

由於是Tensorflow所開發的規格, Labele完畢後的[PASCAL VOC](#) 需要轉換成Record的格式, Hub中已經有提供, 請將[PASCAL VOC](#)與圖片丟置對應資料夾, 並運行“**fasterRcnn.ipynb**”並運行裡面提供的轉換程式即可。

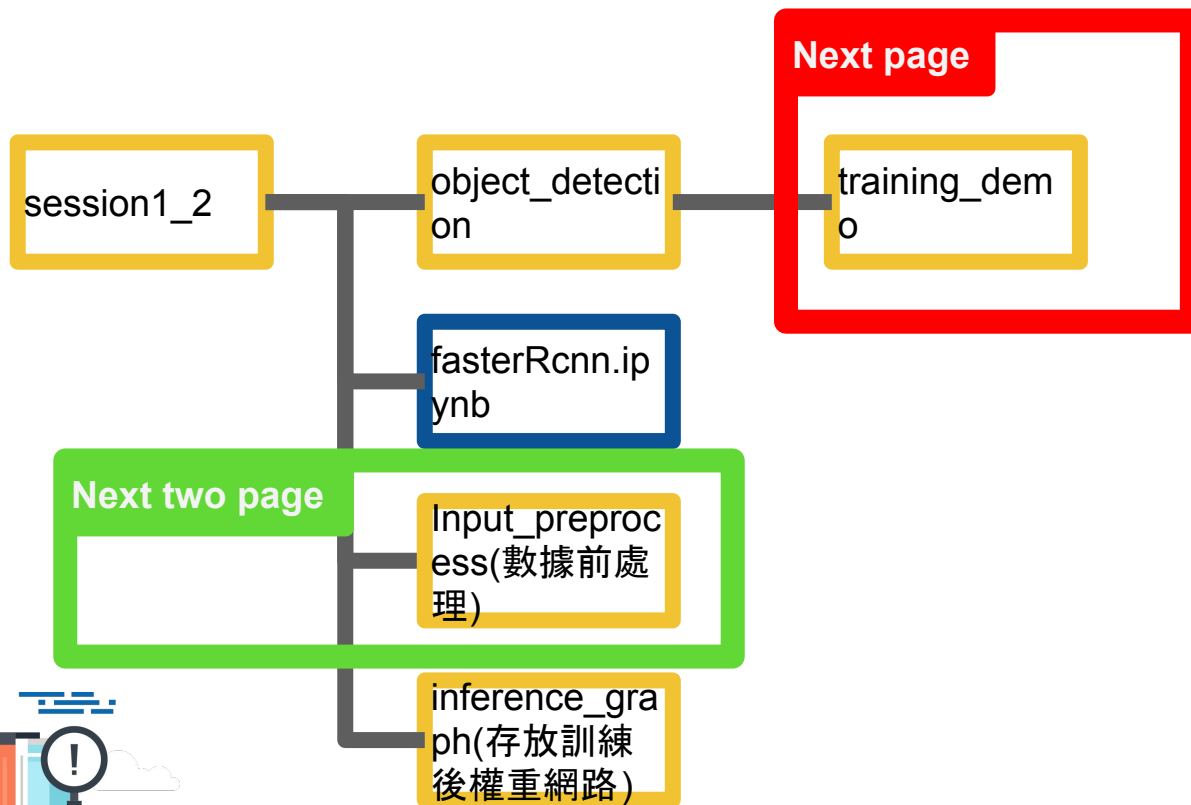


Faster-RCNN手把手 主要步驟

在本篇的程式碼[fasterRcnn.ipynb](#)裡面有說明。



數據放置位置

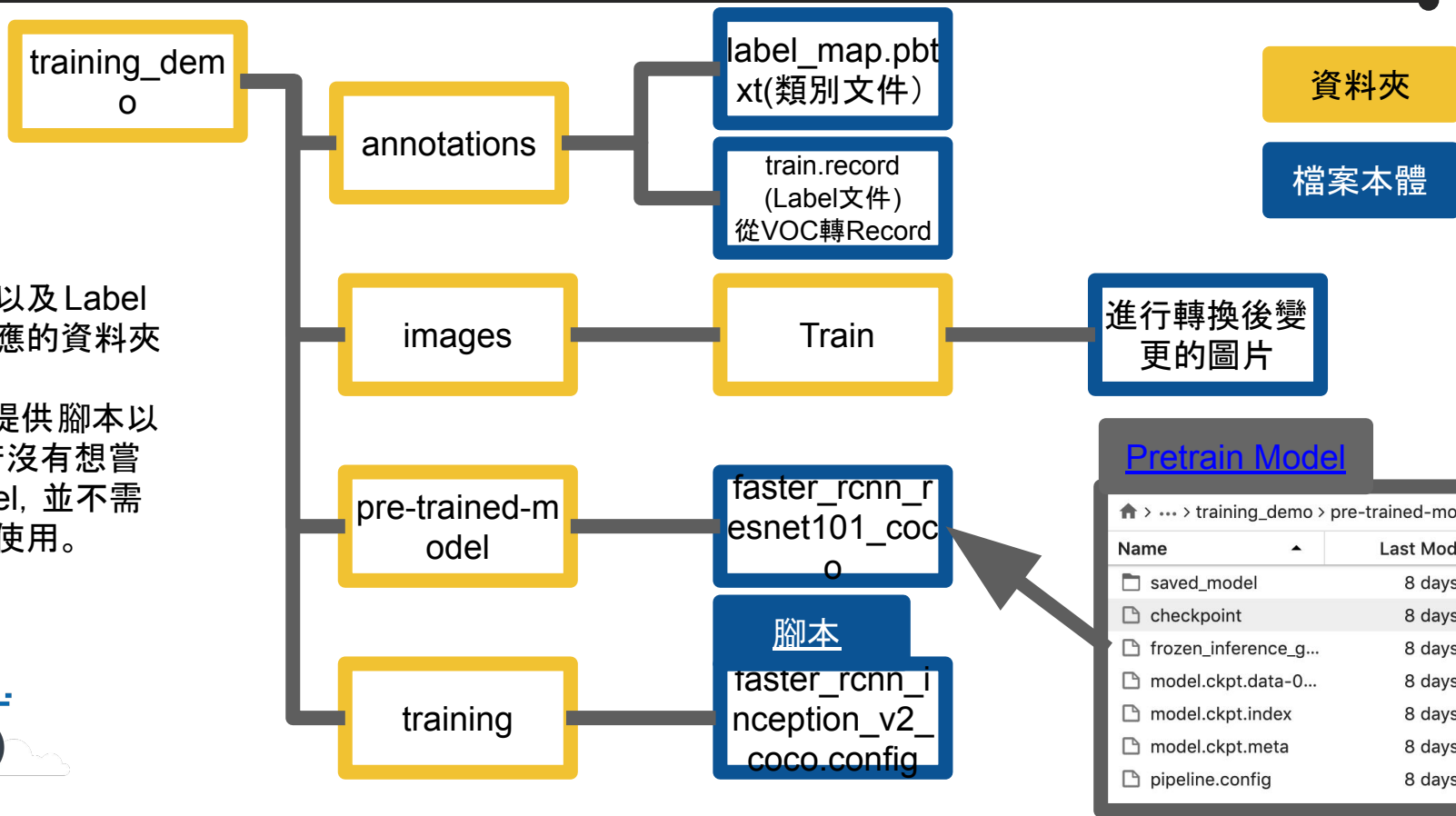


資料夾

檔案本體



數據置放位置



請將資料集以及 Label 文件放置對應的資料夾之下。

Hub已經有提供腳本以及Model, 若沒有想嘗試其他Model, 並不需要額外下載使用。



數據置放位置(數據前處理)

