**homework 3**

1. 流程:
2. 定義模型(第二部份細述)
3. 定義 dataloader(第三部份細述)
4. Dataset前處理
5. (定義train流程) <-- 實際上沒有定義，而是直接執行
6. 定義Plot function
7. 定義結束function
8. 依序執行1~6
9. 使用模型

class ModelBase(keras.Sequential, ABC):

  def \_\_init\_\_(self):

    super().\_\_init\_\_();

  @abstractmethod

  def inputShape(self):

    pass;

  def compile(self):

    # default compile option

    super().compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy']);

  def fit(self, trainSeq, validSeq):

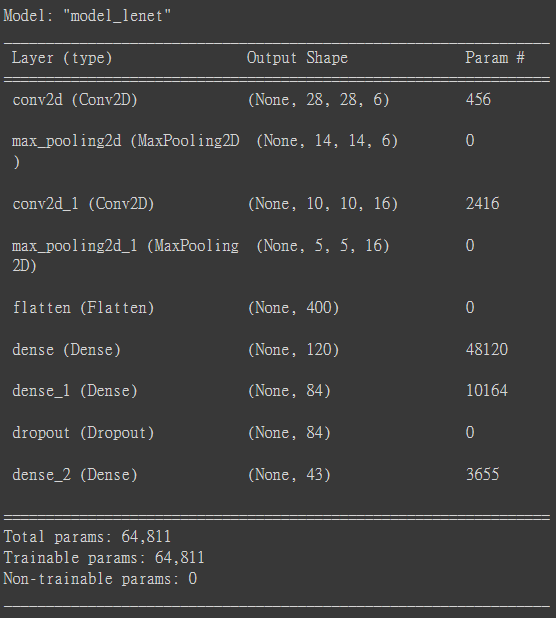
    # override fit option

    return super().fit(x=trainSeq, batch\_size=BATCH\_SIZE, epochs=EPOCHS,

                        verbose=1, validation\_data=validSeq);

基本架構:sequential，optimizer部分使用adam，loss使用cross entropy

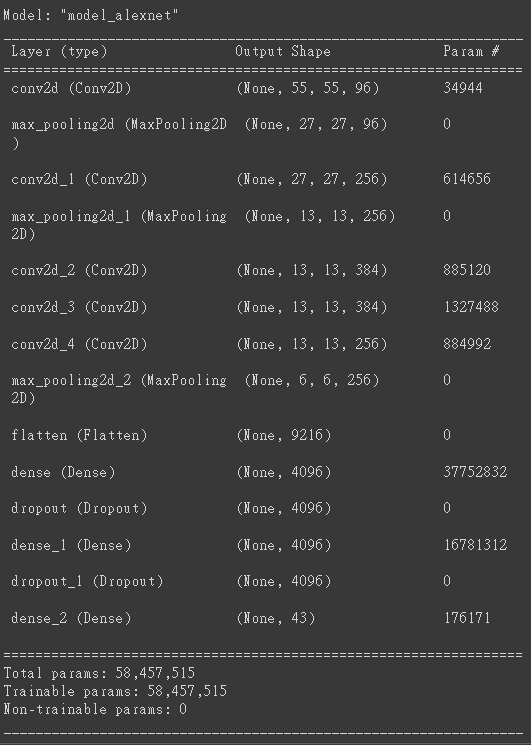
1. Keras的3channel版Lenet5



輸入為32\*32\*3

除了輸出變43外基本沒改任何參數，沒進行任何fine tune作業

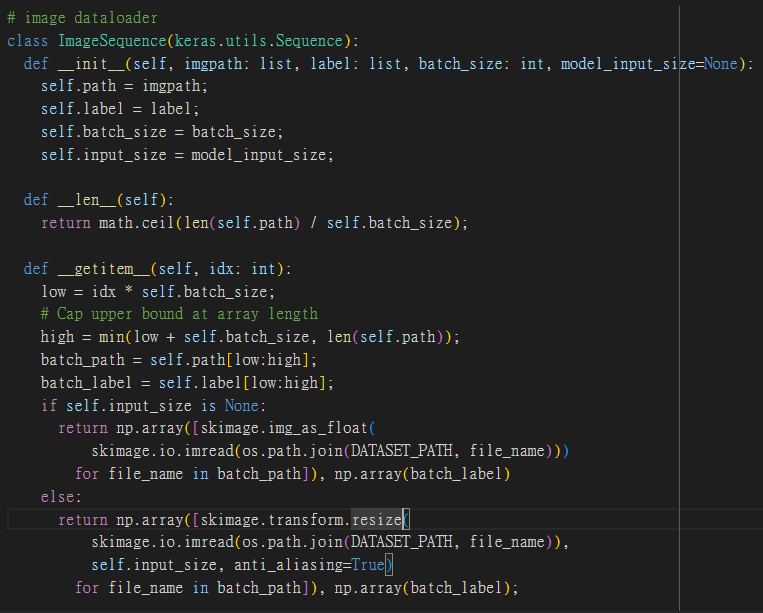
1. Keras的Alexnet



輸入為227\*227\*3

與原本Alexnet不同的是輸出變43及少了batch\_normalization，同時原本分割成上下兩部分的model也變成一個單純sequential的model

1. 使用Dataloader:
2. 使用自訂義keras.utils.Sequence，不同於直接儲存整個dataset，只，這裡使用兩種方案:
3. image dataloader

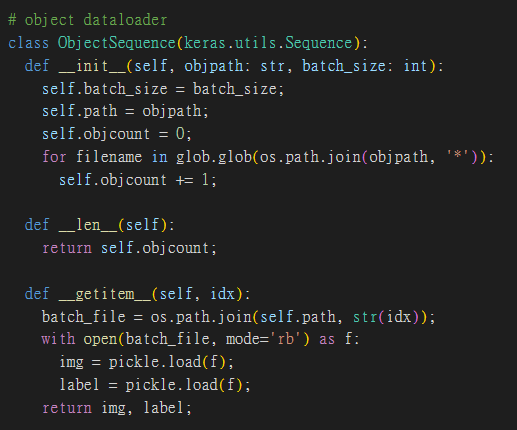


只儲存label及image的路徑，詳細image 等到取index的時候再取得

優點是非常輕便，硬碟需求極低

缺點是resize和read都使用cpu，速度極慢，而且速度與模型輸入大小正相關，以lenet為例，使用object dataloader一個epoch只需要12秒，使用image dataloader則需要55秒，而alexnet

1. object dataloader



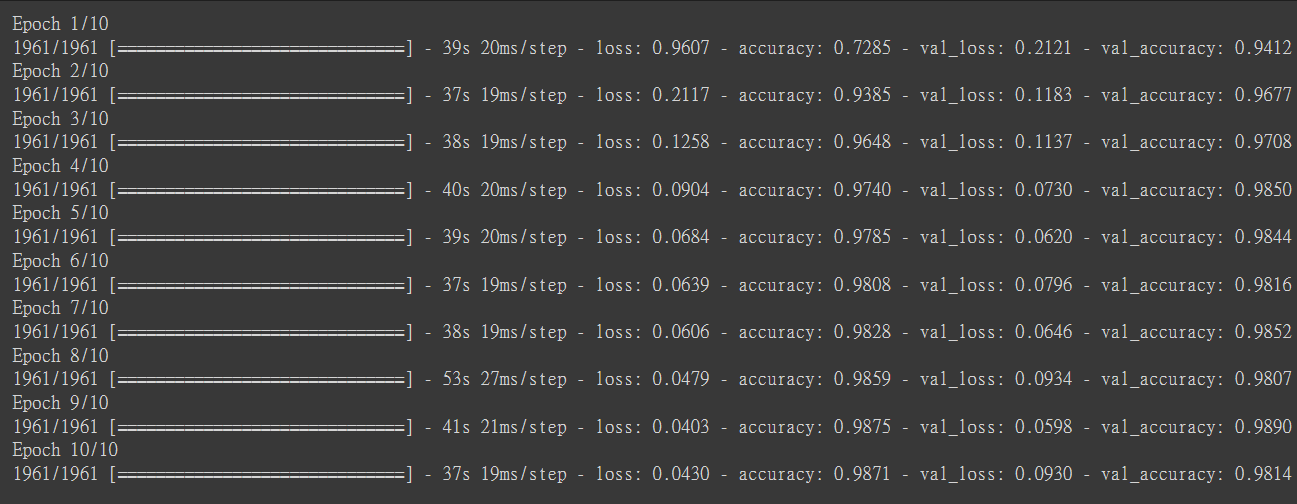
儲存label及object所在folder的位置，需要事先resize並pickle成object檔

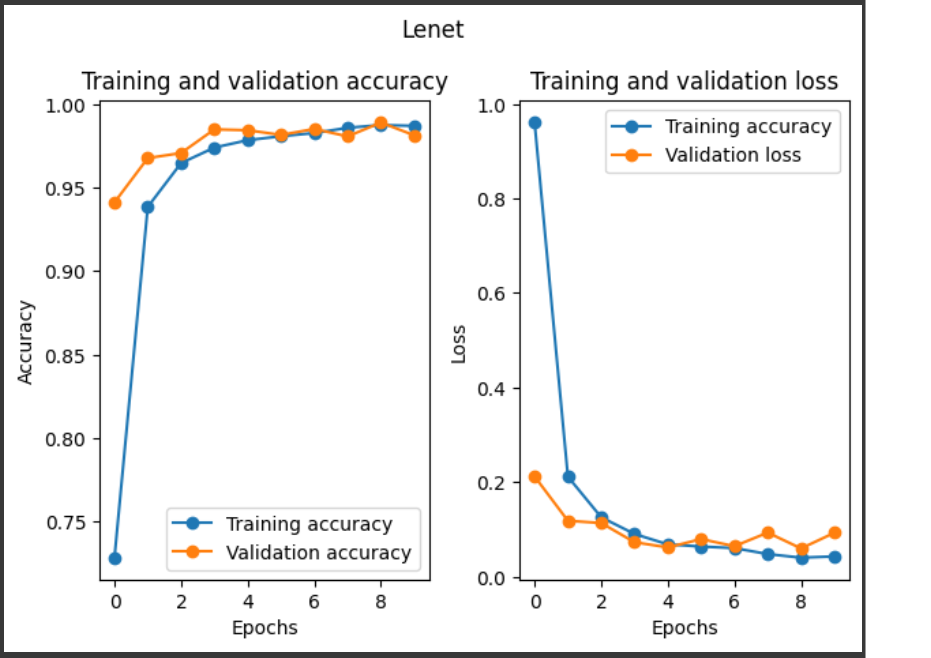
優點是花費時間只比將dataset存進記憶體慢一點。

缺點是非常浪費硬碟空間，dataset及model input越大，所需硬碟空間越多，因此無法完全依賴此方法。

1. 結果

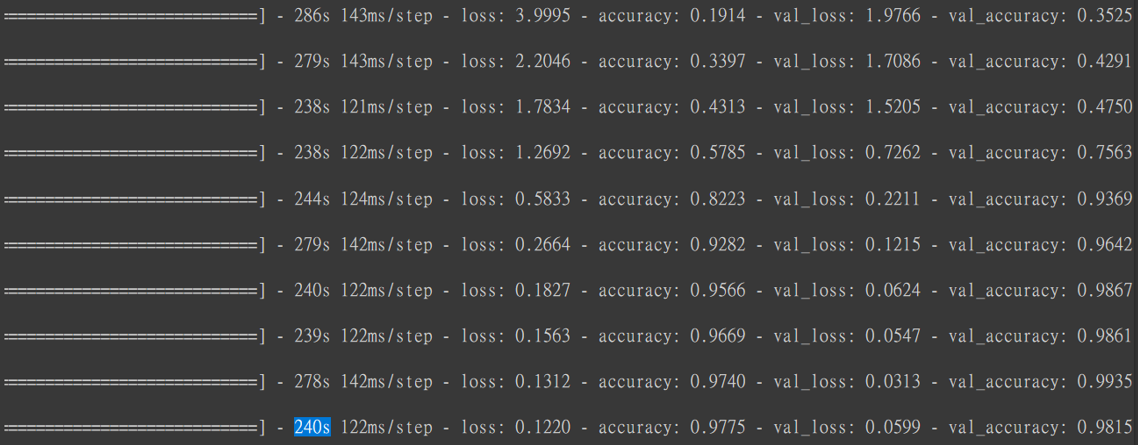
Lenet:

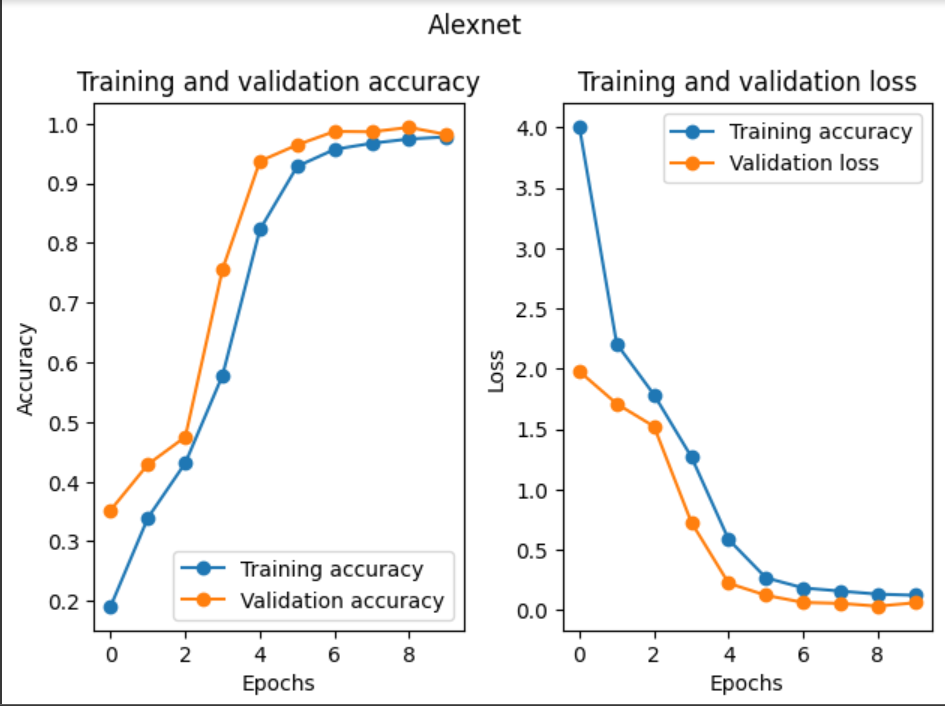


(右邊藍色應為Training loss)

可以發現validation accuracy前期就比training高不少，且accuracy收斂在大約98.5%。推測是keras的dropout在validation階段不會作用。

Alexet:



(右邊藍色應為Training loss)

可以發現validation accuracy前期還是比training高，且accuracy收斂在大約98.5%，收斂時間也較lenet慢。

1. 未來可能的改善:
2. 使用data augmentation讓輸入資料更多元:

考量alexnet光是前處理就花大約10分鐘，且轉換成的object file已經將近40gb，對於colab電腦而言硬碟已接近極限，必須使用其他dataloader。

1. 使用平行及動態dataloader:

Object dataloader在更大的dataset及model input下肯定行不通，因此一定要部分或全部使用image dataloader，但是training過程一下只使用CPU一下只使用GPU非常浪費時間及效能，因此若能training時同時使用其他cpu thread快取圖片進記憶體或硬碟，速度會快一些。

1. 使用更複雜的model:

像是VGG之類的model還好辦，若像是GoogLeNet之類需要平行的model，Tensorflow沒有預設函式庫，這意味必須自己寫平行layer的實作。

1. 各式連結

檔案github: <https://github.com/kc71486/autonomous_hw3>