

NCU AI & ML HW1 ARTCNN

資管二 A 109403019 鄒翔宇

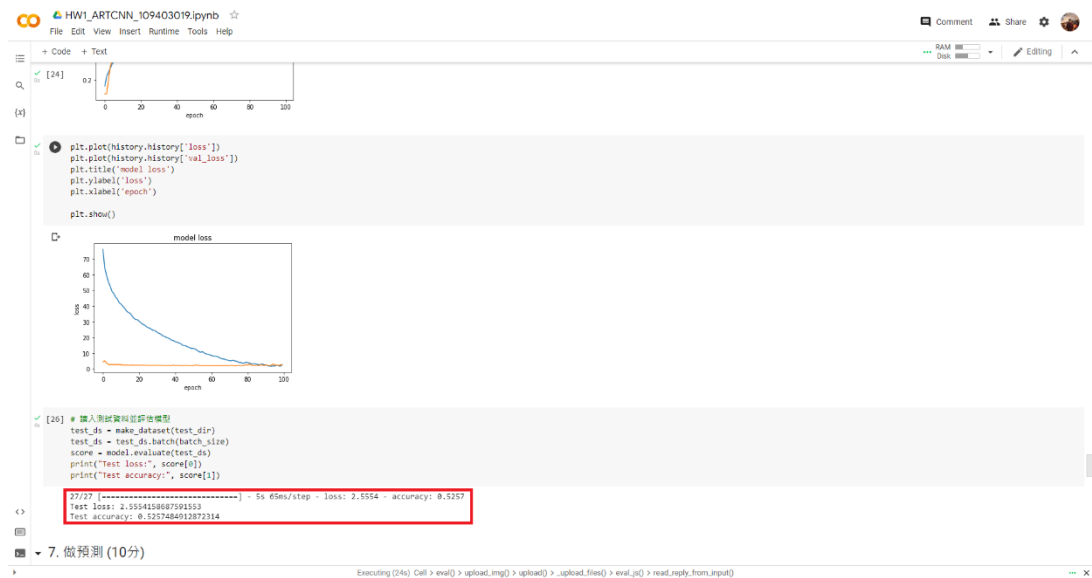
A. HW COLAB LINK :

<https://colab.research.google.com/drive/16iNgKCnbDVAG0cMKd3sXHnQx8vuXJvPr?usp=sharing>

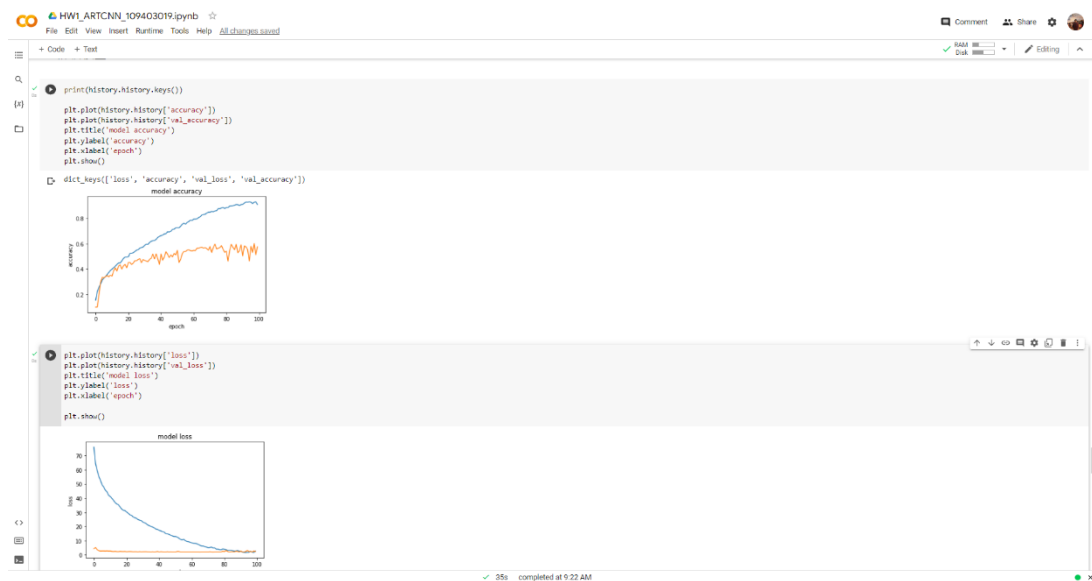
B. DEMO :

epochs=32, batch_size=100

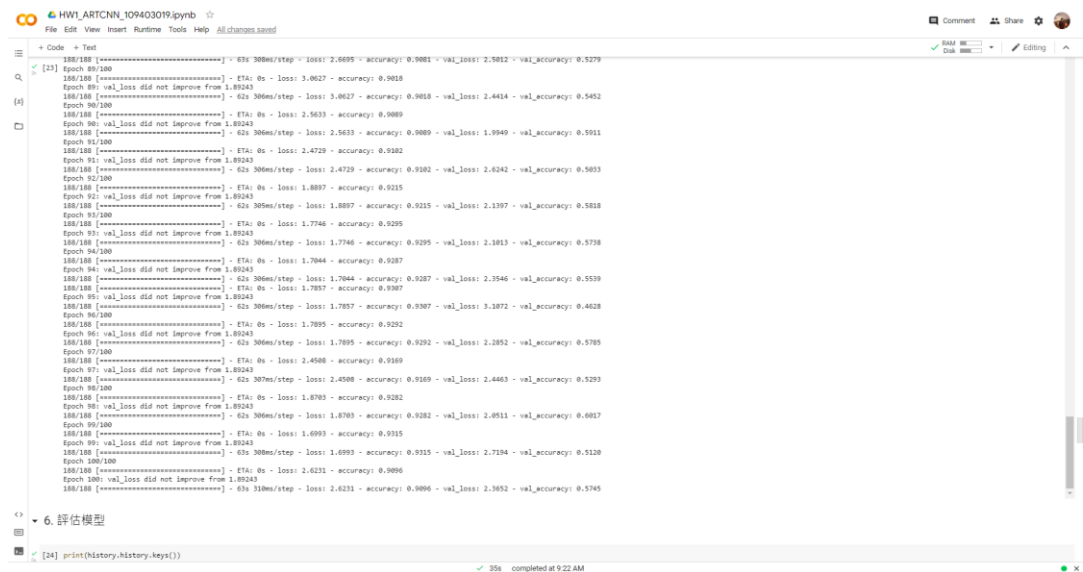
Test Accuracy



Model Accuracy



Training



```
HW1_ARTCNN_109403019.ipynb
File Edit View Insert Runtime Tools Help All changes saved

[23] Epoch 89/100
100/100 [=====] - ETA: 0s - loss: 2.6695 - accuracy: 0.9081 - val_loss: 2.5012 - val_accuracy: 0.5279
Epoch 89: val_loss did not improve from 1.89243
100/100 [=====] - ETA: 0s - loss: 3.0627 - accuracy: 0.9018 - val_loss: 2.4414 - val_accuracy: 0.5452
Epoch 90/100
100/100 [=====] - ETA: 0s - loss: 2.5633 - accuracy: 0.9089
Epoch 90: val_loss did not improve from 1.89243
100/100 [=====] - ETA: 0s - loss: 2.5633 - accuracy: 0.9089 - val_loss: 1.9949 - val_accuracy: 0.5911
Epoch 91/100
100/100 [=====] - ETA: 0s - loss: 2.4729 - accuracy: 0.9382
Epoch 91: val_loss did not improve from 1.89243
100/100 [=====] - ETA: 0s - loss: 2.4729 - accuracy: 0.9382 - val_loss: 2.6242 - val_accuracy: 0.5033
Epoch 92/100
100/100 [=====] - ETA: 0s - loss: 1.8897 - accuracy: 0.9225
Epoch 92: val_loss did not improve from 1.89243
100/100 [=====] - ETA: 0s - loss: 1.8897 - accuracy: 0.9225 - val_loss: 2.1397 - val_accuracy: 0.5818
Epoch 93/100
100/100 [=====] - ETA: 0s - loss: 1.7746 - accuracy: 0.9295
Epoch 93: val_loss did not improve from 1.89243
100/100 [=====] - ETA: 0s - loss: 1.7746 - accuracy: 0.9295 - val_loss: 2.1813 - val_accuracy: 0.5738
Epoch 94/100
100/100 [=====] - ETA: 0s - loss: 1.7844 - accuracy: 0.9287
Epoch 94: val_loss did not improve from 1.89243
100/100 [=====] - ETA: 0s - loss: 1.7844 - accuracy: 0.9287 - val_loss: 2.3546 - val_accuracy: 0.5539
Epoch 95/100
100/100 [=====] - ETA: 0s - loss: 1.7857 - accuracy: 0.9307
Epoch 95: val_loss did not improve from 1.89243
100/100 [=====] - ETA: 0s - loss: 1.7857 - accuracy: 0.9307 - val_loss: 3.1872 - val_accuracy: 0.4628
Epoch 96/100
100/100 [=====] - ETA: 0s - loss: 1.7895 - accuracy: 0.9292
Epoch 96: val_loss did not improve from 1.89243
100/100 [=====] - ETA: 0s - loss: 1.7895 - accuracy: 0.9292 - val_loss: 2.2852 - val_accuracy: 0.5785
Epoch 97/100
100/100 [=====] - ETA: 0s - loss: 2.4588 - accuracy: 0.9389
Epoch 97: val_loss did not improve from 1.89243
100/100 [=====] - ETA: 0s - loss: 2.4588 - accuracy: 0.9389 - val_loss: 2.4463 - val_accuracy: 0.5293
Epoch 98/100
100/100 [=====] - ETA: 0s - loss: 1.8783 - accuracy: 0.9282
Epoch 98: val_loss did not improve from 1.89243
100/100 [=====] - ETA: 0s - loss: 1.8783 - accuracy: 0.9282 - val_loss: 2.0511 - val_accuracy: 0.6817
Epoch 99/100
100/100 [=====] - ETA: 0s - loss: 1.6993 - accuracy: 0.9315
Epoch 99: val_loss did not improve from 1.89243
100/100 [=====] - ETA: 0s - loss: 1.6993 - accuracy: 0.9315 - val_loss: 2.7194 - val_accuracy: 0.5120
Epoch 100/100
100/100 [=====] - ETA: 0s - loss: 2.6231 - accuracy: 0.9086
Epoch 100: val_loss did not improve from 1.89243
100/100 [=====] - ETA: 0s - loss: 2.6231 - accuracy: 0.9086 - val_loss: 2.3652 - val_accuracy: 0.5745

6. 評估模型
[34] print(history.history.keys())
35% completed at 9:22 AM
```

C. 撰寫過程：

基本上皆照著助教的 todo 步驟撰寫。

Pre 資料集下載：

避免每次訓練都需要上傳或是佔用學術網路資源，因此使用同學存在

Google Drive 上訓練資料的的檔案。

1. 讀入封包：無修改。

2. 取得資料集：

因為每個畫家之間的畫作數量很不平均，會造成 class 資料不平衡的問題，並影響到模型的訓練。透過計算權重 (class_weight) 以便後續訓練模型時使用 (資料越多權重越低)

3. 資料前處理：

a. 要將作者名稱作為 class name，因此把作者英文名字與數字做 dict

(class_name) 以方便訓練，這部分先將作者名稱作為 key，並設定一個

index 作為 value。而我們需要 reverse 的結果以方便提取使用，這部分將

先前完成的 class_name 的 value 設為新 dict (rev_class_name) 的 key，key

為 value，便完成了。

- b. 要從各畫作的路徑提取 label (作者名稱)，可以發現到要提取作者名稱只需要取出檔案名稱最後一個 '_' 以前的字串便是作者名稱。接著透過前面已完成的 author_Dict 返回該作者映射的數字
- c. 字串無法運算，須將 label 轉成離散值。而 label 是作者名稱映射的結果，本身的數字並沒有順序意義，所以要將 labels 轉成 onehot，使用 `keras.utils.to_categorical` 便可以達成。
- d. 因為資料集中的照片大小不一定相等，因此要決定輸入模型的圖片長寬並將圖片重新固定大小 (這邊設(256, 256))以及調整解析度至 [0,1]。
- e. 要將資料集中的打散，避免一些相同作者的資料過於集中影響模型訓練結果，此參數設 1500。

4. 建立模型：

- a. 使用 `keras.Sequential` 建立模型，設定好 input shape 後就可以開始做 convolution (激活函式為 relu) 以及 maxpooling，這裡重複做四輪，每一輪都會使用 batch normalization 加速模型收斂，以減緩梯度消失的問題 (維持輸出平均值接近 0 以及輸出標準差接近 1)。
- b. 對整個二維輸入進行 average pooling
- c. 使用兩層 dropout/dense 以提升準確率。dropout 一定比例的 neuron，避免 overfitting 的狀況。

5. 制定訓練計畫：

- a. epochs 原先設為 50，觀察結果認為還有準確率提升的空間，因此調整為 100。
- b. optimizer 使用 Adam 並將 learning rate 設為 0.0001 (雖然 training 時收斂速度較快，但 testing 的誤差也較大)。
- c. 創一個 callback 以保存模型權重。
- d. 使用 `model.fit` 並將 train 的資料、epochs、class weight、validation、callbacks 等參數丟入開始訓練。

6. 評估模型：無修改。

7. 做預測：

將預測作家的模型完成。先拓展輸入 `img` 的維度，使用 `model.predict` 並找到最大值，也就是找出最有可能的作家。因為前面訓練是用數字存入作家，因此要取得作家名稱就要用 `rev_class_name` (數字映射作家名稱的 dict) 並將其存入 `authorName` 回傳。

D. 心得：

這次的作業是我第一次接觸到 machine learning 實作，也是第一次使用 colab，覺得新奇也很有趣，可以把課堂上學習到的知識實作出來。不過也因為是第一次接觸，因此一開始完全不知道要如何起手，不停地上網查資料以及詢問同學。

過程中一些簡單的如合併路徑、映射作者名稱與數字等可以非常容易快速處理。而有些如圖片大小、batch size、epochs 決定等則是透過測試許多次觀察趨勢去做修正。資料前處理的部分問題不大。最有趣的肯定還是建置模型並訓練的過程，因為可以透過嘗試不同的參數，得到不一樣的結果，雖然訓練過程相當耗時，但是可以看到不同的趨勢就覺得非常有趣，若發現準確率有所提升或是 loss 下降心情就會很好。

建立模型最大的突破是將 Convolution 以及 MaxPooling 將兩輪提升至四輪。只做兩輪 (batch_size=32, epochs=50) 的 test accuracy 只有 0.36 左右，距離理想的至少 0.5 還有一大段距離，而提升到四輪 (batch_size=32, epochs=50) 的 test accuracy 則來到了 0.48，既然趨勢有提升，看起來也尚未有 overfitting 的狀況，因此只要再將 epochs 做適當的增加想必能獲得更高的準確度。不過尷尬的是每次只要做完 50 輪基本上就會被鎖 GPU，掛機又會被斷線，因此到了快截止時才訓練出 100 輪的結果，而也非常開心 test

accuracy 來到 0.526，從一開始的 0.36 到 0.526，這樣的進步讓我感到相當有成就感，也讓我更了解了 CNN 實作的過程。不過想必還是有很大進步空間，待學習更多 CNN 的知識後勢必會再回來提升本次作業的模型準確率。

REFERENCES :

1. [DeepArtist : Identify Artist from Art](#)
2. [使用 class weight 和 sample weight 处理不平衡问题](#)
3. <https://stackoverflow.com/questions/483666/reverse-invert-a-dictionary-mapping>
4. https://keras.io/api/layers/normalization_layers/batch_normalization/
5. <https://medium.com/ching-i/batch-normalization-%E4%BB%8B%E7%B4%B9-135a24928f12>
6. https://colab.research.google.com/github/tensorflow/docs-110n/blob/master/site/zh-cn/tutorials/keras/save_and_load.ipynb?hl=zh-cn#scrollTo=mQF_dlgIVovq