404410030 資工三 鄭光宇

HW2

Method description:

利用 VGG16 模型，來對 CroppedYale 人臉資料進行分類。

1. 將資料集中，每一種人臉資料**隨機**取 35 張訓練資料，30 張測試資料。
2. 將資料集輸入到 VGG16 進行訓練，以**隨機、預訓練兩種方式**對權重初始化，分別對這兩種模型都比較**使用 Data Augmentation、不使用 Data Augmentation**，總共比較**四種**設定的模型，還有作業一 Nearest Neighbor 方法，最後收斂得到的 accuracy、loss 分別是多少。
3. 使用 **Data Augmentation** (在底下表格中簡成 aug.) 來對抗 overfitting，**增強**模型能力。

Experimental results – accuracy:

以下整理四種情況下，訓練完成時在 test set 的 loss, accuracy：

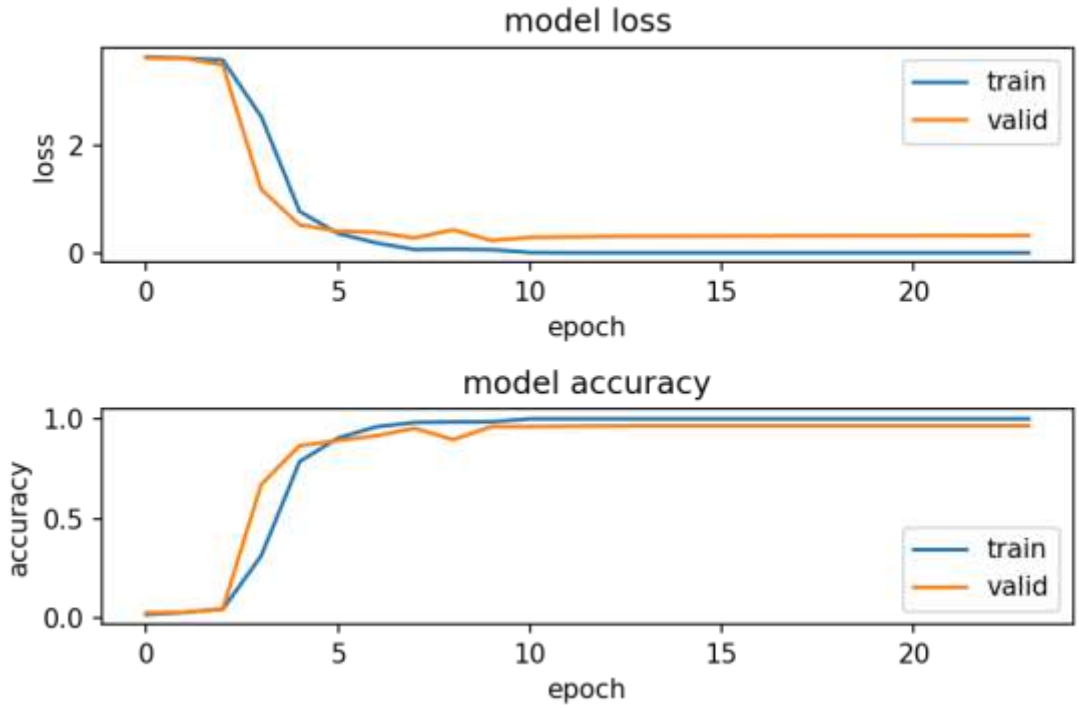
	loss	accuracy
Random weight	0.322	0.965
Random weight + aug.	0.236	0.981
Pre-trained weight	0.203	0.974
Pre-trained weight + aug.	0.678	0.861
Nearest Neighbor (SSD)	N/A	0.459
Nearest Neighbor (SAD)	N/A	0.451

黃色代表最好，紅色代表最差。可以看出，在使用 random 初始化時，若使用 Data Augmentation，可以提升 accuracy；然而若使用助教提供的 pre-trained weight，Data Augmentation 反降低 accuracy，而且模型 loss plot 有相較其他 model 較大的震盪，也許是因為套用 data augmentation 後的 CroppedYale 資料集與 pre-trained weight 的資料集差異太大？讓模型愈學愈糟？

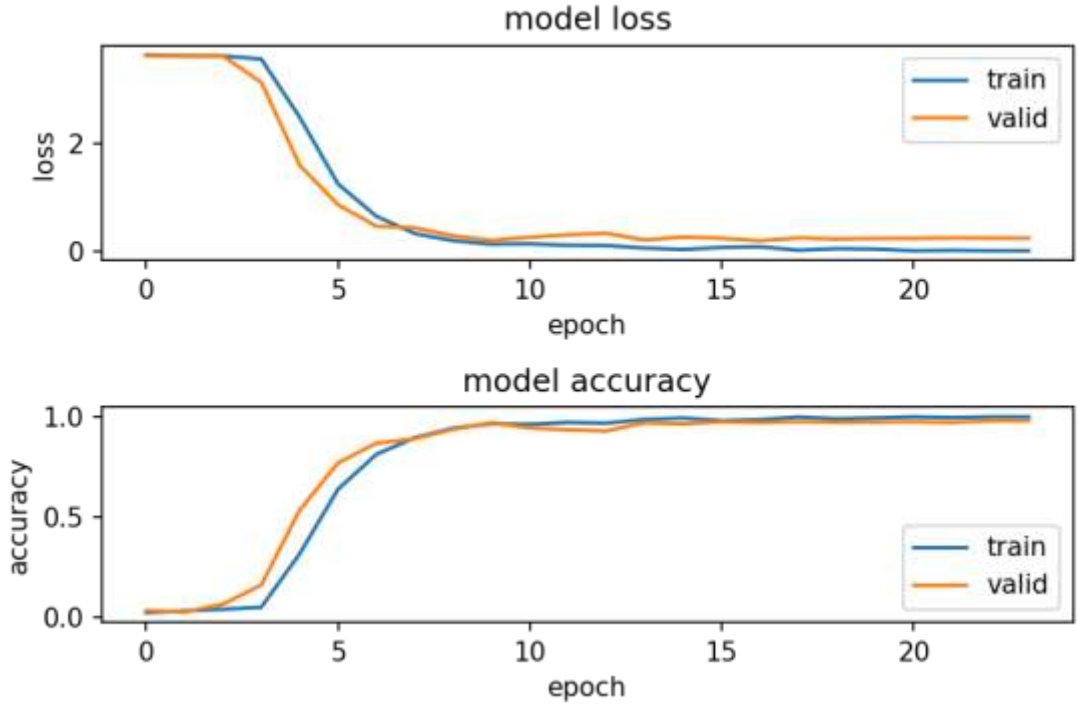
而得到最高 **accuracy** 的模型，是使用 **random weight 與 data augmentation** 的組合。得到最低 loss 的模型，是使用 pre-trained weight 與不使用 data augmentation 的組合。

以下是四個模型訓練的 loss plot:

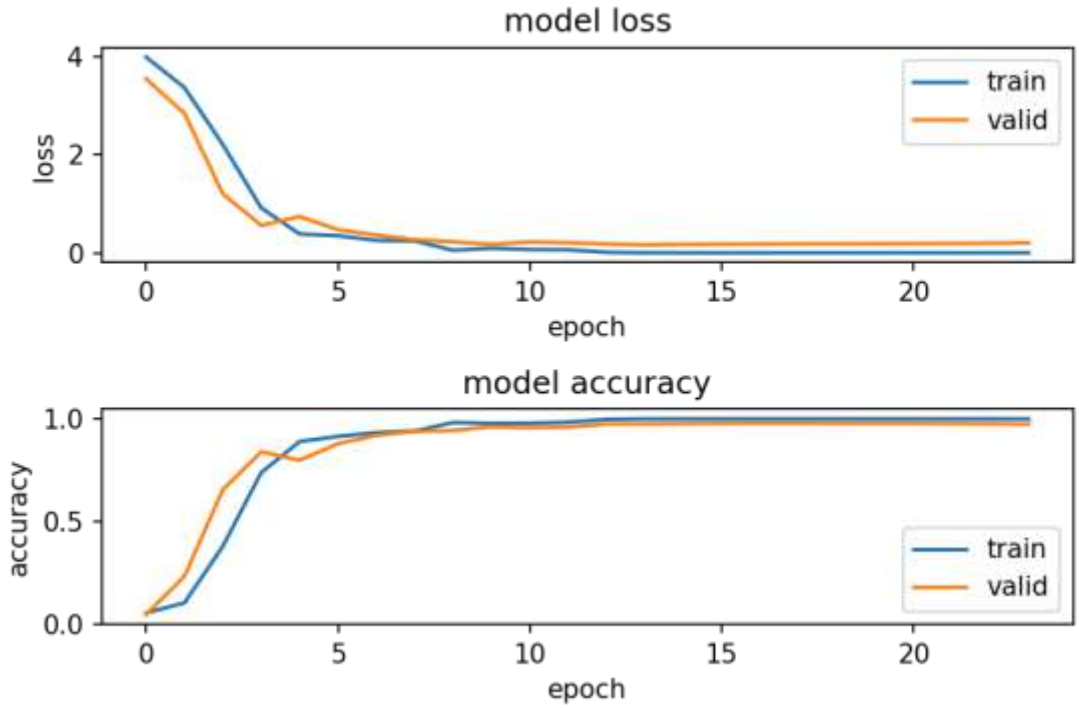
Random weight:



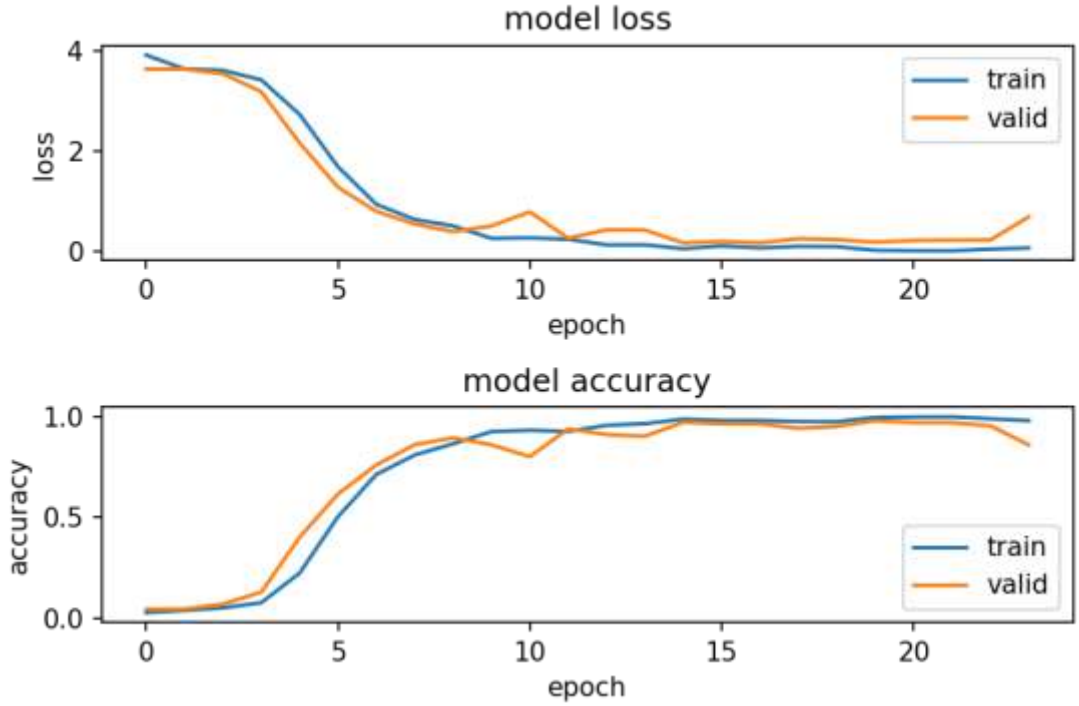
Random weight + aug.:



Pre-trained weight:



Pre-trained weight + aug.:



Discussion of difficulty or problem encountered:

實作上沒什麼問題。不過剛開始看到 vgg16.py 範例檔的時候愣了一下，因為裡面的函式似乎沒有 include_top 這個參數。後來觀察到 vgg16.py 與 keras 官網上的定義是一致的，所以後來直接 import keras 內建的 vgg16 模型來用。