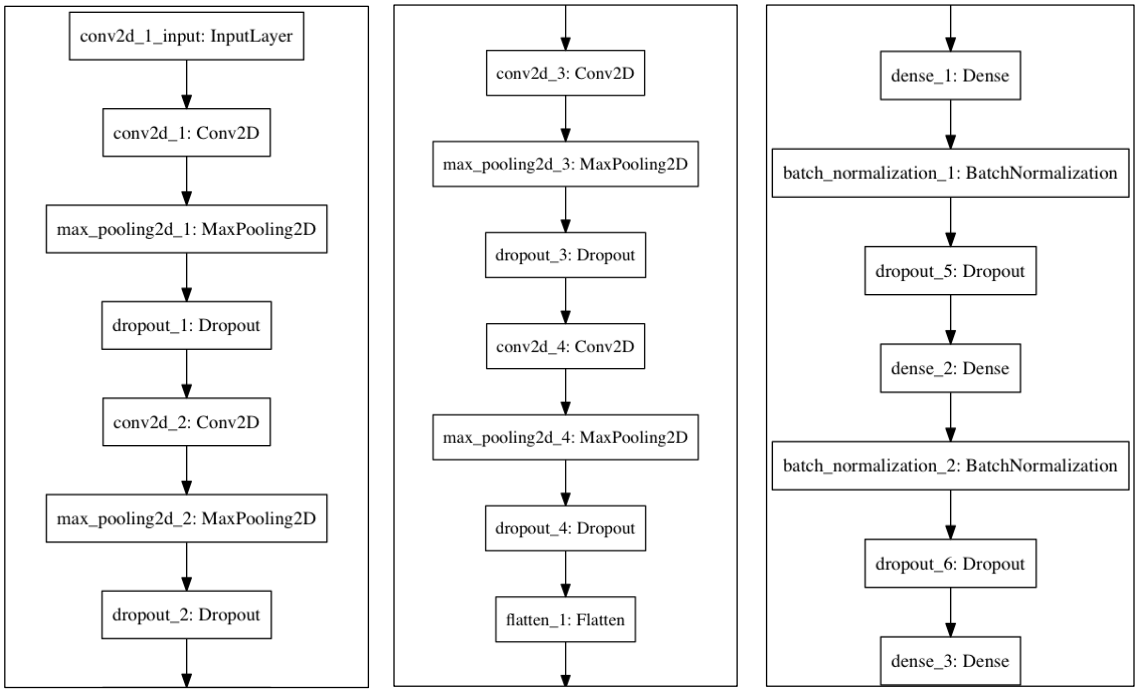
1. (1%) 請說明你實作的 CNN model，其模型架構、訓練過程和準確率為何？
(Collaborators: 曾千育)

1) 模型架構



如上圖所示，訓練資料經過四個由 Conv2D()、MaxPooling2D()、Dropout() 組成的 Convolutional Block，經過 Flatten() 壓平後再過由 Dense()、BatchNormalization()、Dropout() 組成的兩個 Dense Block，最後通過 Softmax 分成 7 類。每層詳細參數：

- Conv2D()：filter 數量依序為 64, 128, 256, 512 個；kernel_size 除了第一層為 (5,5) 外，其餘皆為 (3,3)；padding 皆為 same；activation 皆為 selu；kernel_initializer 皆採用 glorot_normal。
- MaxPooling2D()：pool_size 皆為 (2,2)；padding 為 same。
- Dense()：unit 數量依序為 512, 256 個；activation 皆為 selu；kernel_initializer 皆採用 glorot_normal。
- Dropout()：rate 依序為 0.35, 0.4, 0.45, 0.5, 0.5, 0.45。

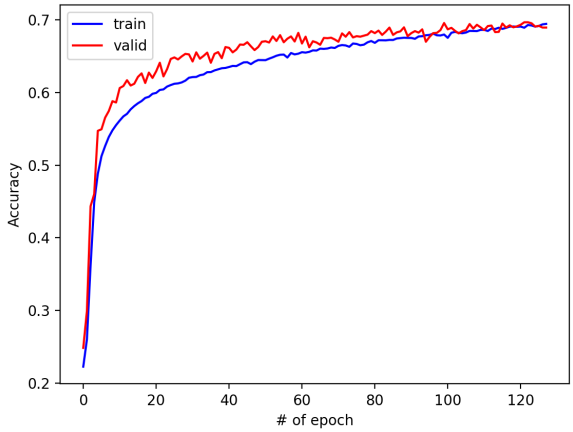
整個 model 架構包含 4,046,855 個參數（其中有 1536 個為 non-trainable）。

2) 訓練過程

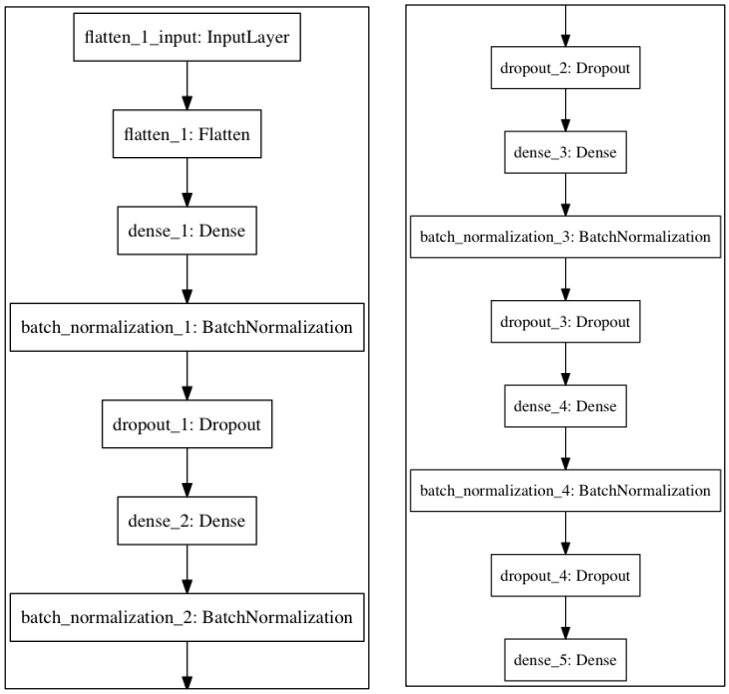
隨機取出訓練資料的 20% 作為 validation set，另外為了增加訓練 model 的準確率，經由 keras 中的 ImageDateGenerator 得到旋轉、平移、縮放及鏡射後的新資料，通過 fit_generator() 來訓練時共 128 個 epoch，每 epoch 會看 137728 筆資料。而 loss function 使用 categorical_crossentropy，並使用 Adam() 作為 optimizer。另外使用 ModelCheckpoint 這個 keras 的 Callback function 來保留準確率較高的 model。

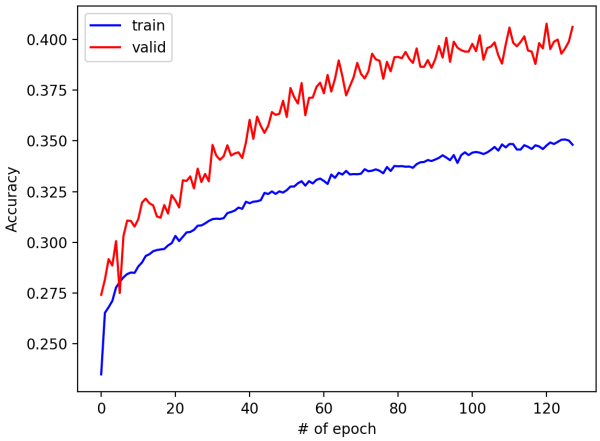
3) **準確率**

右圖為訓練過程中準確率的走勢圖（其中 training accuracy 有被 dropout 影響），可以看出 validation accuracy 在不到 20 個 epoch 就有 60% 的表現，最終可以收斂至約 69%左右。而 Kaggle 上最好的成績是由三個相似的 CNN model 進行 ensemble 得到的，其綜合準確率達到約 70.35%。



2. **(1%) 承上題，請用與上述 CNN 接近的參數量，實作簡單的 DNN model。其模型架構、訓練過程和準確率為何？試與上題結果做比較，並說明你觀察到了什麼？**





1) **模型架構**

如左上圖所示，一開始使用 Flatten() 壓平圖片資料，中間使用了四組 Dense()、BatchNormalization()、Dropout() 的設計，最後通過 Softmax 分七類。詳細參數：

- Dense()：unit 數量依序為 1024, 1024, 512, 256 個；activation 皆為 relu。
- Dropout()：rate 皆為 0.5。

整個 model 架構包含 4,079,111 個參數（其中有 5632 個為 non-trainable）。

2) **訓練過程**：與前一題的 CNN 時幾乎一樣。

3) **準確率**：右上圖為 DNN 訓練過程中的準確率走勢圖，可以看出在經過 128 個 epoch 後，validation accuracy 大約收斂到 40%。

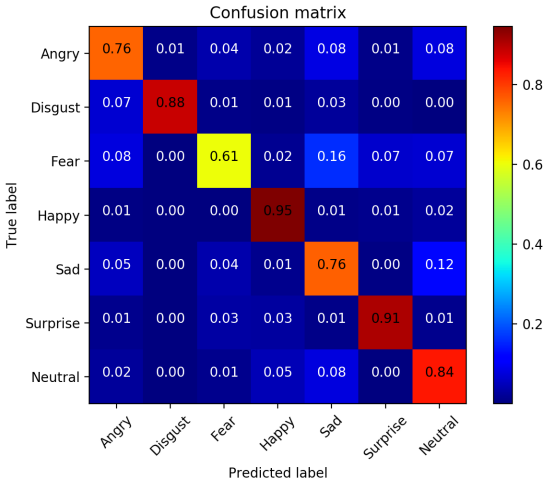
4) **比較**

在準確率上 CNN 有約 70% 的表現，而 DNN 只有 40%，這顯示了 convolution layer 用作於圖像辨識上的效果能帶給我們的差異。另外可以從準確率走勢曲線看出 CNN 的收斂速度較快（不到 20 epoch 就趨近收斂）而 DNN 收斂數度較慢（約 100 epoch 以上

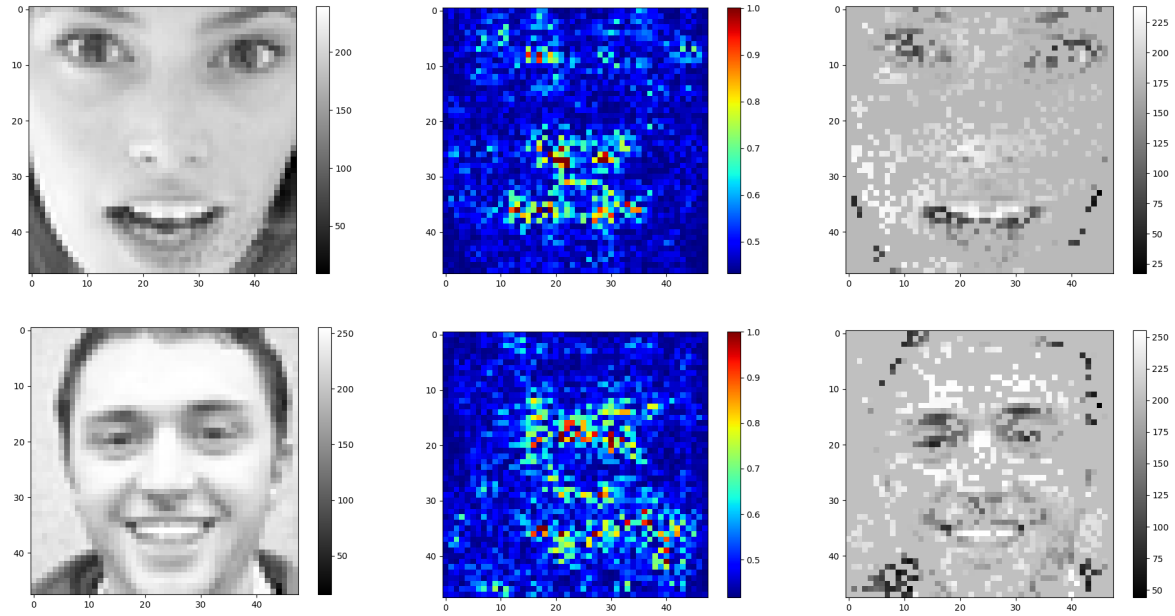
才收斂）。而在訓練 model 的運算速度上，同樣是 128 個 epoch，CNN 需時 3 小時，而 DNN 需不到 1 小時，這顯示增加 convolution layer 所帶來的運算負擔差異。

3. **(1%) 觀察答錯的圖片中，哪些 class 彼此間容易用混？[繪出 confusion matrix 分析]**

由右圖可以看出，最容易用混的情形為，當資料 label 為 Fear 時，我訓練的 CNN model 只有 61% 的準確率，其中將 Fear 誤判為 Sad 的機率最高(16%)。另外在 label 為 Sad 時，準確率只有 76%，其中將 Sad 誤判為 Neutral 的機率最高(12%)；而 label 為 angry 時，準確率相同只有 76%，其中將 Angry 誤判為 Sad 或 Neutral 的機率皆為 8%，這兩個並列第二容易用混的情形。

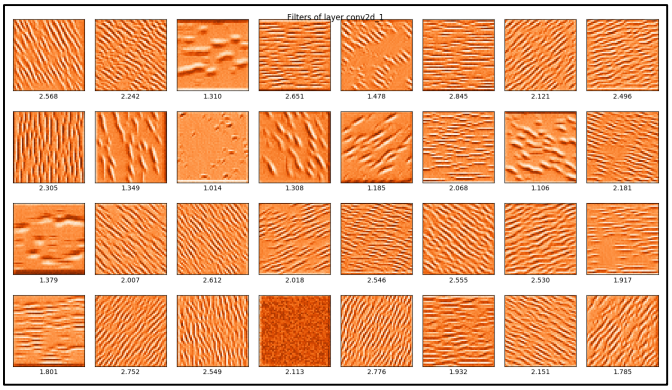


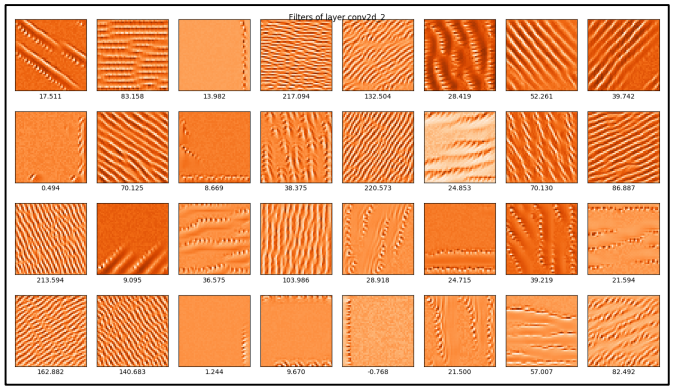
4. **(1%) 從(1)(2)可以發現，使用 CNN 的確有些好處，試繪出其 saliency maps，觀察模型在做 classification 時，是 focus 在圖片的哪些部分？**

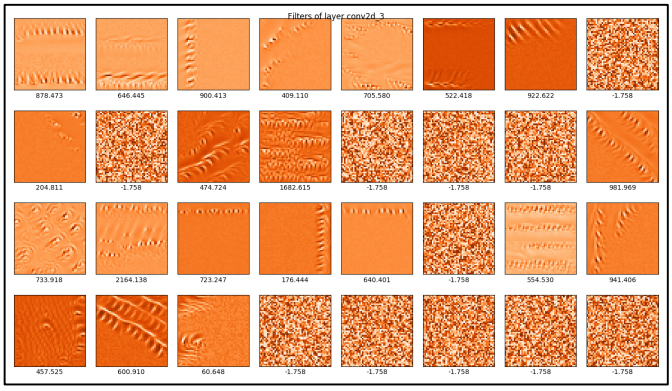


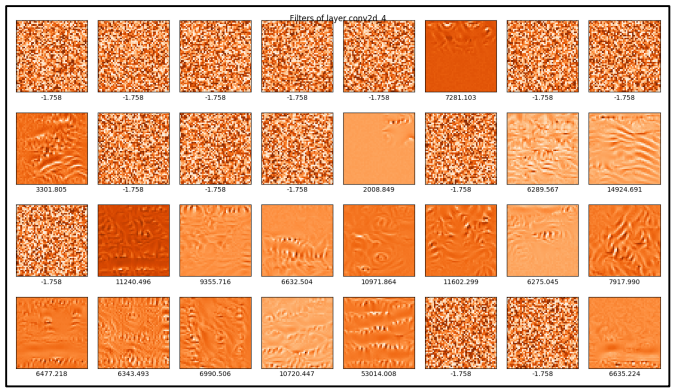
由上圖可以看出 heatmap 中，有在眼睛、嘴巴、鼻子等地方有較高的值，由右邊經過較高的值的 mask 出來的圖片可以看出這些地方確實也是我們在判斷這兩個人表情的主要依據。

5. **(1%) 承(1)(2)，利用上課所提到的 gradient ascent 方法，觀察特定層的 filter 最容易被哪種圖片 activate。**



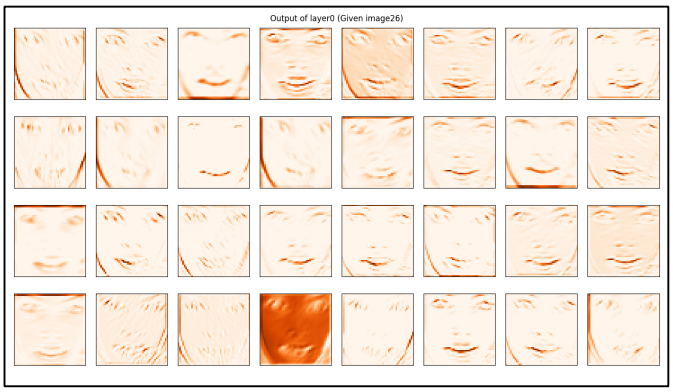


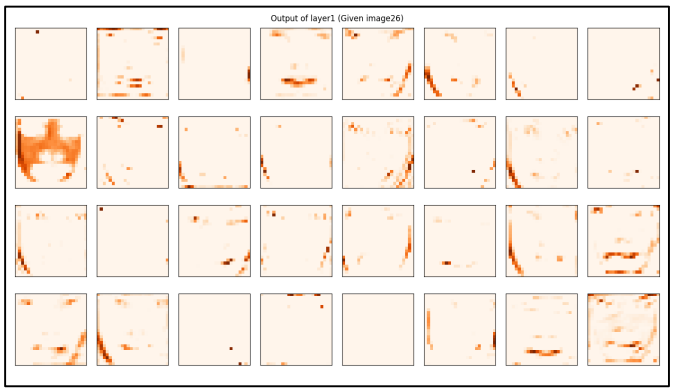




上面四張圖為在四個 convolution layer 中，透過 gradient ascent，畫出理論上能夠最大化 output 的圖形（每個 layer 只取了其中 32 個 filter 作圖）。可以看出較淺層的 filter（左上圖）表現出的都是各種線條與塊狀的 filter，而隨著 layer 越走越深，filter 的樣子開始變得可以描繪更具體的樣子，像是各種圈圈有如在描述眼睛、鼻子等部位，而各種波紋有可能是在描述嘴巴等部位。

另外在深層的 filter 中有出現一些看起來完全是雜訊的 filter，猜測有兩種可能性，第一是機器理解到我們人類無法理解的特徵，所以這些 filter 可能看似雜訊，其實對機器是有用的；第二種可能是由於 filter 數量過多或過深，導致產生出許多較無用的 filter。





上面兩張圖為將某張圖輸入，取 32 個 filter 輸出的結果。上左圖為較淺層的 filter 輸出的結果，可以看出其 filter 篩選描繪出整個人臉的樣子；而上右圖為較深層的 filter 輸出的結果，可以看出已經有取得更進一步的細微的部位，像是有篩選出嘴巴、眼睛、甚至是將五官挖掉的特殊圖形。

備註：

由於整份報告中有許多不同的部分有跟許多人討論過，難以辨別是哪一題，故一起列在最後：曾千育、陳弘梵、劉瀚聲、林政豪。