

學號：B04611041 系級：工海三 姓名：簡暉晉

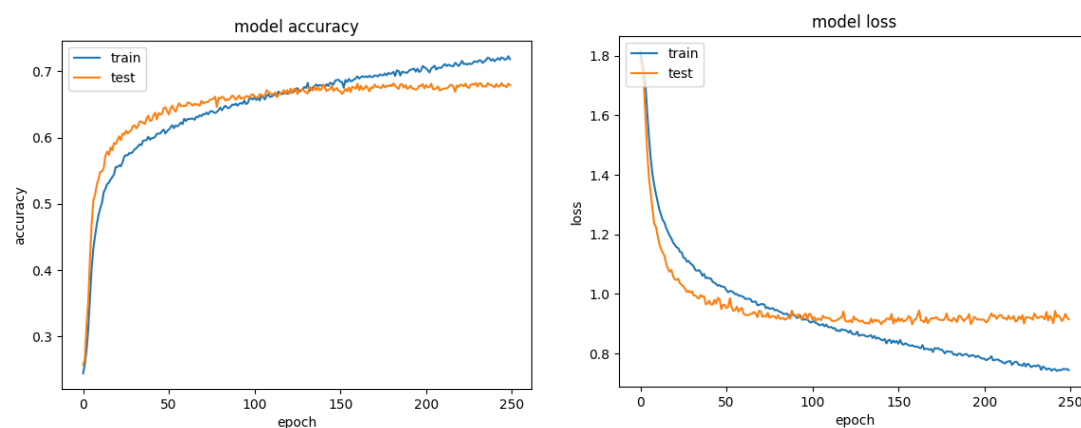
1. (1%) 請說明你實作的 CNN model, 其模型架構、訓練過程和準確率為何?
(Collaborators: No)

model 架構:

| Layer (type) | Output Shape | Param # | | | |
|--------------------------------|---------------------|---------|--------------------------------|---------------------|---------|
| | | | conv2d_4 (Conv2D) | (None, 12, 12, 256) | 295168 |
| conv2d_1 (Conv2D) | (None, 48, 48, 64) | 640 | activation_4 (Activation) | (None, 12, 12, 256) | 0 |
| activation_1 (Activation) | (None, 48, 48, 64) | 0 | max_pooling2d_3 (MaxPooling2D) | (None, 6, 6, 256) | 0 |
| conv2d_2 (Conv2D) | (None, 48, 48, 64) | 36928 | dropout_3 (Dropout) | (None, 6, 6, 256) | 0 |
| activation_2 (Activation) | (None, 48, 48, 64) | 0 | flatten_1 (Flatten) | (None, 9216) | 0 |
| max_pooling2d_1 (MaxPooling2D) | (None, 24, 24, 64) | 0 | dense_1 (Dense) | (None, 1024) | 9438208 |
| dropout_1 (Dropout) | (None, 24, 24, 64) | 0 | activation_5 (Activation) | (None, 1024) | 0 |
| conv2d_3 (Conv2D) | (None, 24, 24, 128) | 73856 | dropout_4 (Dropout) | (None, 1024) | 0 |
| activation_3 (Activation) | (None, 24, 24, 128) | 0 | dense_2 (Dense) | (None, 7) | 7175 |
| max_pooling2d_2 (MaxPooling2D) | (None, 12, 12, 128) | 0 | activation_6 (Activation) | (None, 7) | 0 |
| dropout_2 (Dropout) | (None, 12, 12, 128) | 0 | | | |
| | | | Total params: 9,851,975 | | |
| | | | Trainable params: 9,851,975 | | |
| | | | Non-trainable params: 0 | | |

我使用 4 層 Convolution 的 layer, conv2d_1, conv2d_2, conv2d_3, conv2d_4, filter 數目分別為 64, 64, 128, 256, activation function 皆使用 relu, 並做 zero padding(padding=same), 其中 2~4 層之後做了 dropout 和 maxpooling, , pooling size 皆為 2x2, 最後的 dense layer 用一層 1024 neuro 的 layer, activation function 為 relu, 最後 output 7 個 neuron, activation function 使用 softmax。

訓練過程:



我取出 training set 的 20% 作為 validation set, 並且做了 data augmentation, 將圖片在 training 時作平移、旋轉等, 除了增加 data 外也可以增加 noise, 可以減少 overfitting。而 optimizer 則使用 adamax, loss function 為 cross-entropy。最後跑 250 個 epoch。而 kaggle 上的最佳結果則是由 adam 和 adamax 這兩個 optimizer 並且用所有 training data 最後做 ensemble 的結果。由上圖的訓練過程可以看出一開始因為增加了 noise 所以 training 的 accuracy 反而比 val_acc 低, 而過了約 100 個 epoch 後 val_acc 的上升則逐漸趨緩。最終單一 model kaggle 上的準確率為 0.69406, ensemble 後為 0.70298, 可發現做 ensemble 後也達到了不錯的效果。

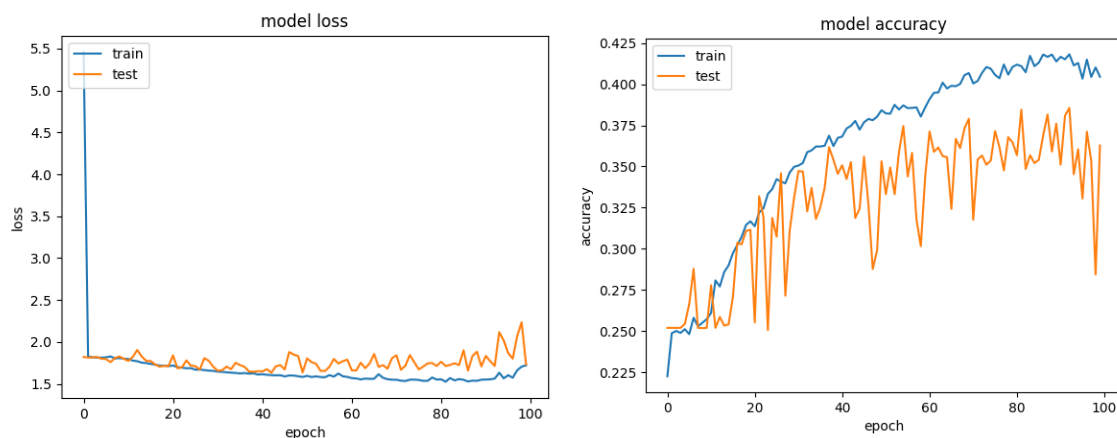
2. (1%) 承上題，請用與上述 CNN 接近的參數量，實做簡單的 DNN model。其模型架構、訓練過程和準確率為何？試與上題結果做比較，並說明你觀察到了什麼？

(Collaborators:No)

架構：

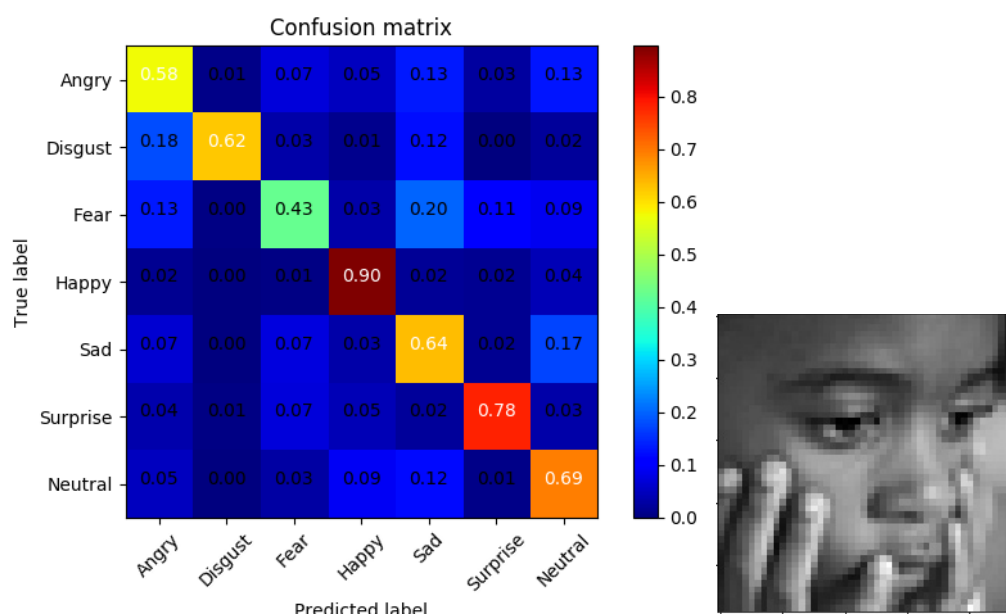
| Layer (type) | Output Shape | Param # | | | |
|---------------------|--------------|---------|-----------------------------|--------------|---------|
| dense_1 (Dense) | (None, 512) | 1180160 | dense_6 (Dense) | (None, 1024) | 1049600 |
| dense_2 (Dense) | (None, 512) | 262656 | dropout_5 (Dropout) | (None, 1024) | 0 |
| dropout_1 (Dropout) | (None, 512) | 0 | dense_7 (Dense) | (None, 1024) | 1049600 |
| dense_3 (Dense) | (None, 1024) | 525312 | dropout_6 (Dropout) | (None, 1024) | 0 |
| dropout_2 (Dropout) | (None, 1024) | 0 | dense_8 (Dense) | (None, 1024) | 1049600 |
| dense_4 (Dense) | (None, 1024) | 1049600 | dropout_7 (Dropout) | (None, 1024) | 0 |
| dropout_3 (Dropout) | (None, 1024) | 0 | dense_9 (Dense) | (None, 1024) | 1049600 |
| dense_5 (Dense) | (None, 1024) | 1049600 | dropout_8 (Dropout) | (None, 1024) | 0 |
| dropout_4 (Dropout) | (None, 1024) | 0 | dense_10 (Dense) | (None, 1024) | 1049600 |
| | | | dropout_9 (Dropout) | (None, 1024) | 0 |
| | | | dense_11 (Dense) | (None, 7) | 7175 |
| | | | Total params: 9,322,503 | | |
| | | | Trainable params: 9,322,503 | | |
| | | | Non-trainable params: 0 | | |

訓練過程：



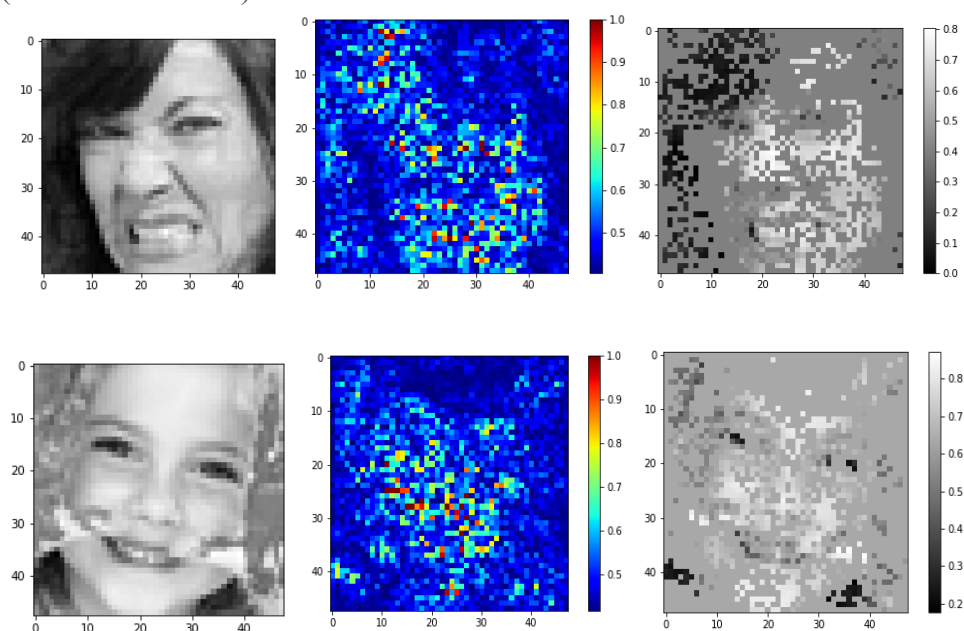
在參數量差不多的情況下，dnn 的準確率只有 36%左右(kaggle)，與 cnn 相差非常多，推測有可能是因為 dnn 不像 cnn 有 filter 可以過濾圖片的特徵，但 dnn 的收斂速度非常快，loss 在幾個 epoch 後就沒有太大的變化，至於 training acc 則大約在 41%就上不去了

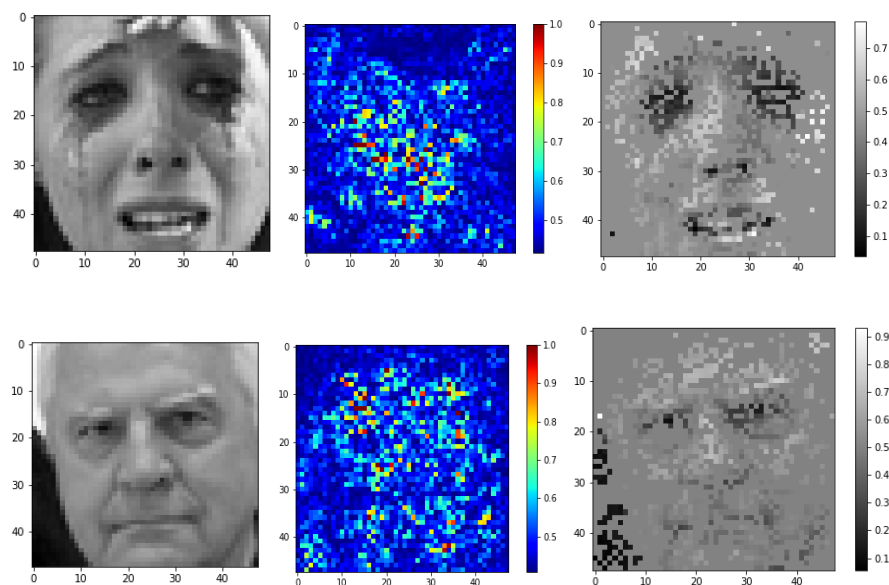
3. (1%) 觀察答錯的圖片中，哪些 class 彼此間容易用混？[繪出 confusion matrix 分析]
(Collaborators:No)



從左圖 confusion matrix 中可以發現，我的 model 在判斷 Fear 時容易判定成 sad，disgust 則容易判定成 angry，而 sad 容易判定成 neutral，另外這幾個 class 也會有一些互相混淆的情形。經由觀察發現這些判定錯誤的 class 大多集中在負面情緒，正面情緒的 Happy 和 surprise 是判定比較準確的。推測可能是因為負面情緒有較多的共同特徵，界定較模糊，因此讓 model 不易判別。而實際觀察了幾張圖片，發現這幾個 class 界定的方式的確比較有爭議。比如右圖雖然正確答案為 fear，但我認為歸類為 sad 其實也未必不行。

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 在圖片的哪些部份？(Collaborators:No)

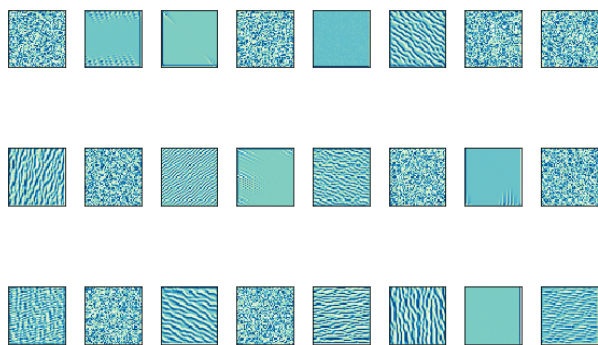




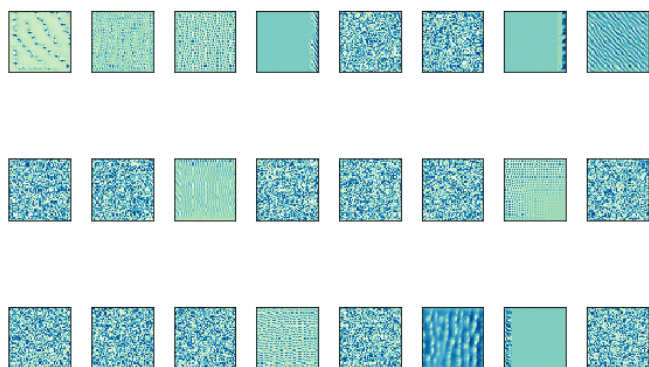
我隨機選了 4 種不同的 class 畫出 **saliency map**，發現 cnn model 在預測時大多 focus 在眼睛，和嘴巴等部分，而這也大概是人類判斷表情的主要依據，因此推斷 cnn 的確成功掌握了人臉辨識的要點。

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**。
(Collaborators:No)

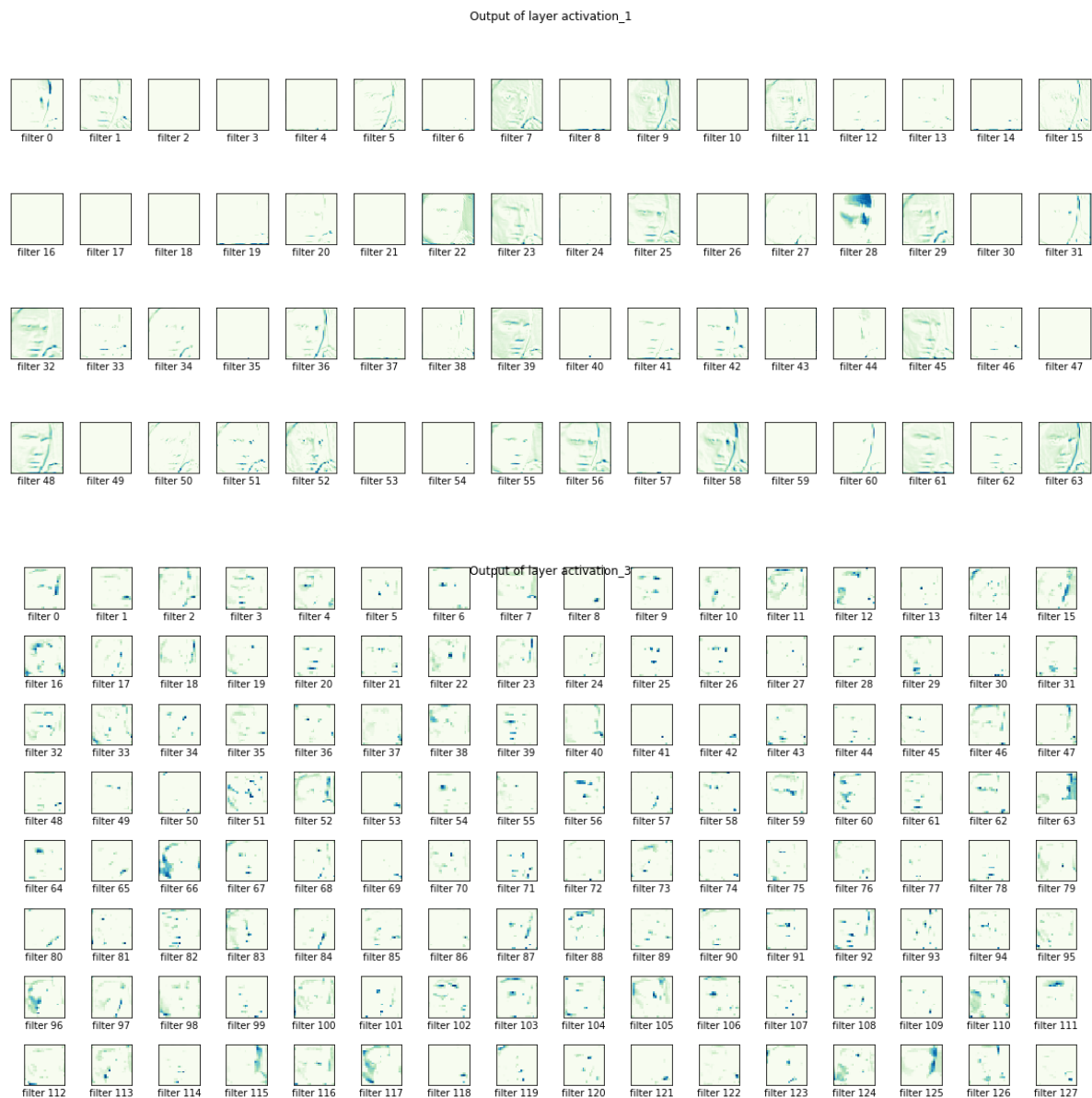
Conv2d_1 filter :



Conv2d_3 fileter :



上圖是我用 gradient ascent 跑 300 個 epoch 後得結果，分觀察了 Conv2d_1 和 Conv2d_4 當中的幾個 filter，發現大多紋路好像都看不太出來有什麼意義，不過有幾張 filter 看起來有一些圓圓的紋路，有可能是在偵測眼睛或嘴巴之類的器官。



上面兩張圖是輸入一張圖片，看 filter 會 activate 圖片中哪些部分，可以發現前面 layer(conv2d_1 經過 activation)的 filter 有很多都可以過濾出人臉的輪廓，而後面的 layer(conv2d_3 經過 activation)的 filter 因為經過 maxpooling 後變的比較難辨識，但仍可看出大部分都 focus 在眼睛和嘴巴等器官上。