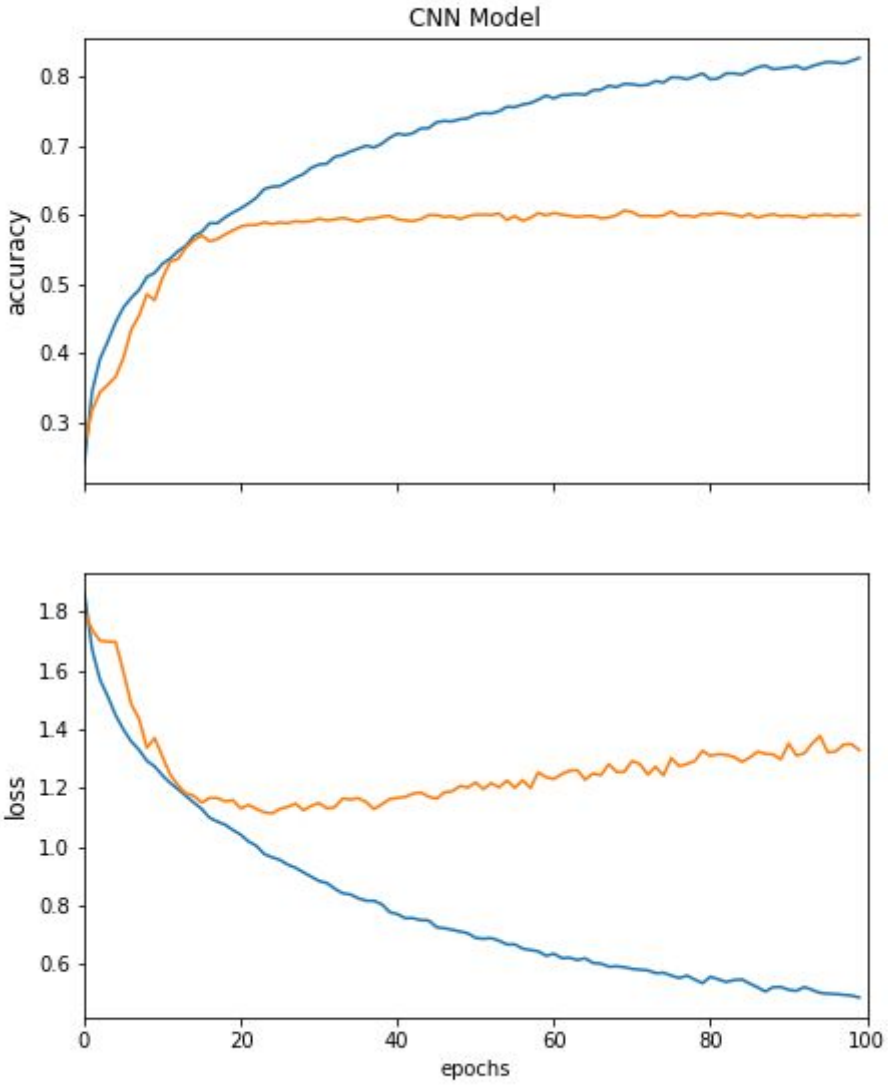
學號：R05229014 系級： 大氣碩二 姓名：鄒適文

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: r05229016 羅章碩))

答：



* 藍色線是training_data * 橘色線是validataion_data

model結構如最上面三張圖表示，而為了不要overfitting，我在每個conv層與fc層都加了dropout和batchnormlization，但發現在大概第18個epoch之後val_acc就沒有再下降過了，大概停在0.6左右，而val_loss持續緩緩上升，大概還是有overfitt的狀況發生。

p.s 因為比較早寫report，這並不是我交上去的model，但與第二題是同一個cnn

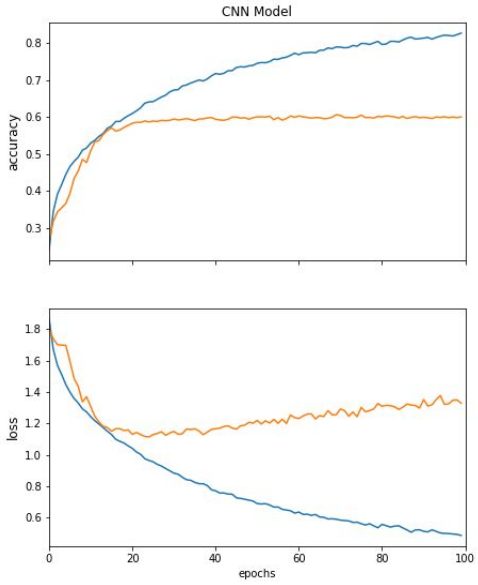
2. (1%) 承上題，請用與上述 CNN 接近的參數量，實做簡單的 DNN model。其模型架構、訓練過程和準確率為何？試與上題結果做比較，並說明你觀察到了什麼？
(Collaborators:)

CNN: Total params: 908,871, Trainable params: 898,567, Non-trainable params: 10,304

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 48, 48, 32)	832
max_pooling2d_1 (MaxPooling2	(None, 24, 24, 32)	0
dropout_1 (Dropout)	(None, 24, 24, 32)	0
batch_normalization_1 (Batch	(None, 24, 24, 32)	128
conv2d_2 (Conv2D)	(None, 24, 24, 64)	18496
max_pooling2d_2 (MaxPooling2	(None, 12, 12, 64)	0
dropout_2 (Dropout)	(None, 12, 12, 64)	0
batch_normalization_2 (Batch	(None, 12, 12, 64)	256
conv2d_3 (Conv2D)	(None, 12, 12, 64)	36928
max_pooling2d_3 (MaxPooling2	(None, 6, 6, 64)	0
dropout_3 (Dropout)	(None, 6, 6, 64)	0
batch_normalization_3 (Batch	(None, 6, 6, 64)	256
conv2d_4 (Conv2D)	(None, 6, 6, 128)	73856
dropout_4 (Dropout)	(None, 6, 6, 128)	0
flatten_1 (Flatten)	(None, 4608)	0
batch_normalization_4 (Batch	(None, 4608)	18432
dense_1 (Dense)	(None, 128)	589952
dropout_5 (Dropout)	(None, 128)	0
batch_normalization_5 (Batch	(None, 128)	512
dense_2 (Dense)	(None, 256)	33024
dropout_6 (Dropout)	(None, 256)	0
batch_normalization_6 (Batch	(None, 256)	1024

dense_3 (Dense)	(None, 512)	131584
dropout_7 (Dropout)	(None, 512)	0
dense_4 (Dense)	(None, 7)	3591

Total params: 908,871
Trainable params: 898,567
Non-trainable params: 10,304



*** 藍色線是**training_data *** 橘色線是**validataion_data

DNN:Total params: 909,255, Trainable params: 902,855, Non-trainable params: 6,400

Layer (type)	Output Shape	Param #
flatten_2 (Flatten)	(None, 2304)	0
batch_normalization_7 (Batch	(None, 2304)	9216
dense_5 (Dense)	(None, 128)	295040
dropout_8 (Dropout)	(None, 128)	0
batch_normalization_8 (Batch	(None, 128)	512

dense_6 (Dense) (None, 256) 33024

dropout_9 (Dropout) (None, 256) 0

batch_normalization_9 (Batch (None, 256) 1024

dense_7 (Dense) (None, 512) 131584

dropout_10 (Dropout) (None, 512) 0

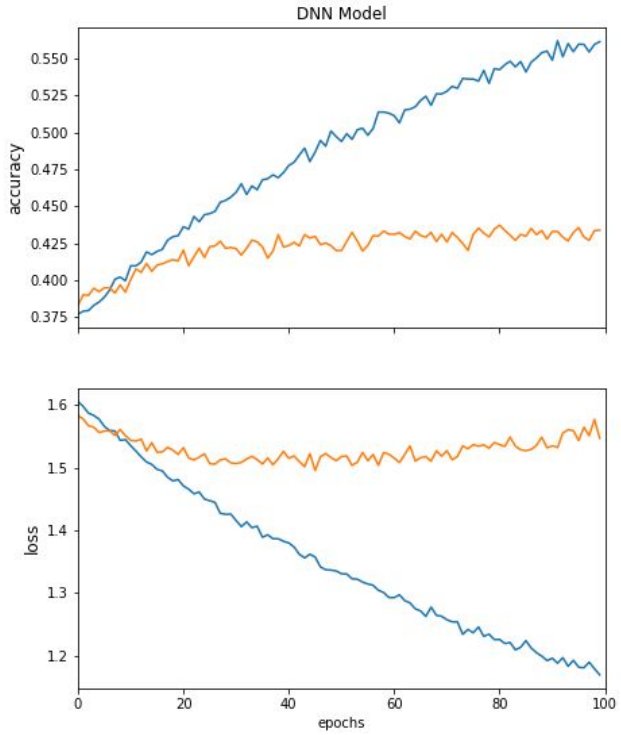
batch_normalization_10 (Batc (None, 512) 2048

dense_8 (Dense) (None, 840) 430920

dropout_11 (Dropout) (None, 840) 0

dense_9 (Dense) (None, 7) 5887

Total params: 909,255
Trainable params: 902,855
Non-trainable params: 6,400

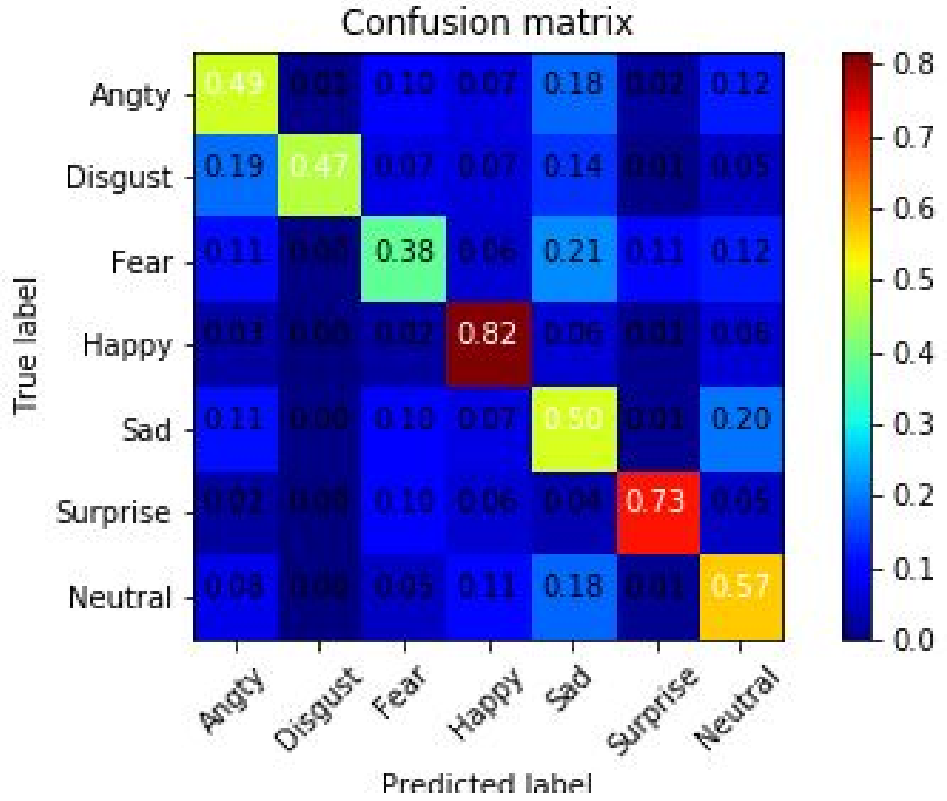


* 藍色線是training_data * 橘色線是validataion_data

設計概念大致與第一題相同，都是為了避免overfitting加了很多dropout跟batchnormalization，而架構如model.summary()所示，並且調整參數量稍微大於cnn

但可以很明顯地看到dnn表現遜於cnn，val_acc大概在0.425就上不去了，同時訓練的線較不平滑，有很多的上下抖動，所以大概可以很明顯推論cnn的conv層可以很好的提取某些特徵出來，同時透過pooling壓縮圖片，可以達到較有效率的模型。

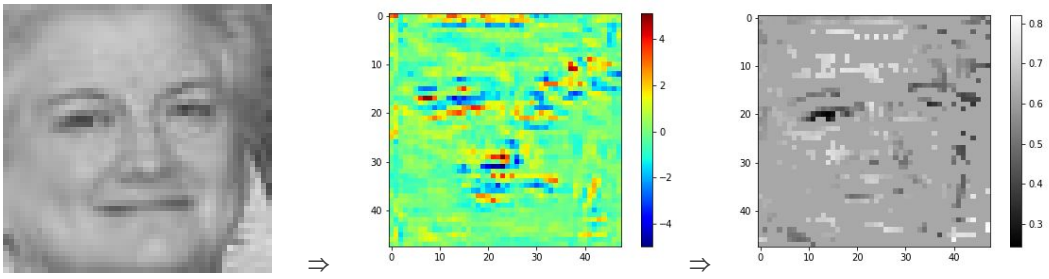
3. (1%) 觀察答錯的圖片中，哪些 class 彼此間容易用混？[繪出 confusion matrix 分析]
(Collaborators:)



答：在我的model中，angry容易被認成sad；disgust容易被認成angry；fear容易被認成sad(最糟的表現)；happy不太容易被認錯(最好的表現)；sad容易被認成neutral；suprise容易被認成fear(次好表現，其實也不太容易被認錯)，neural則容易被認成sad，

以上結論，sad與neural容易被我的model搞混，而以人的認知來說，較為誇張(表情較大)的表情的辨別正確率較高。

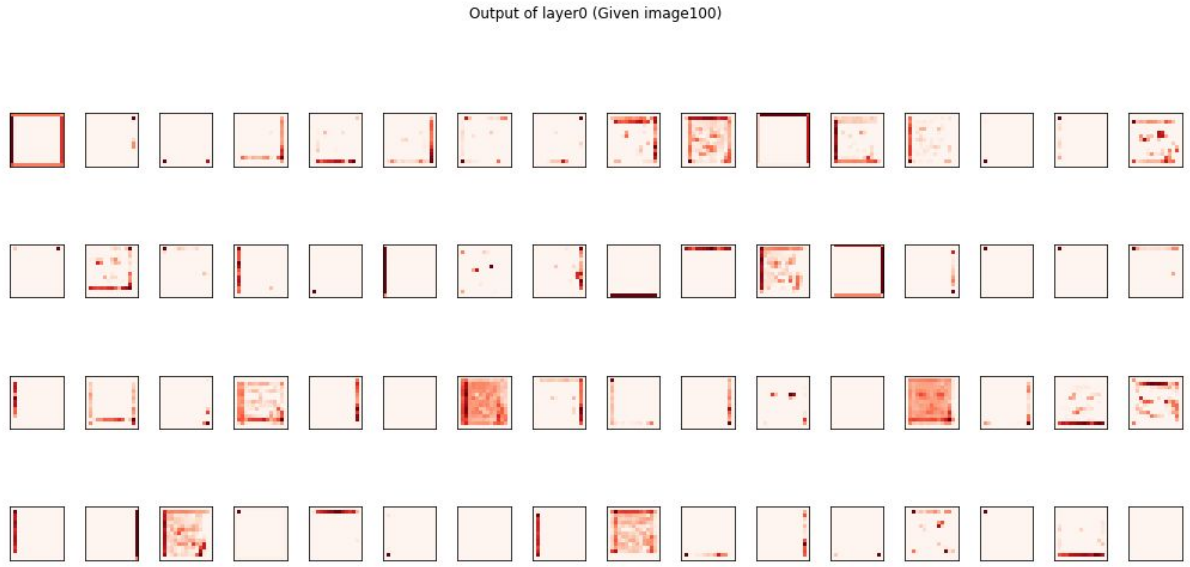
4. (1%) 從(1)(2)可以發現，使用 CNN 的確有些好處，試繪出其 saliency maps，觀察模型在做 classification 時，是 focus 在圖片的哪些部份？
(Collaborators: r05229016 羅章碩)

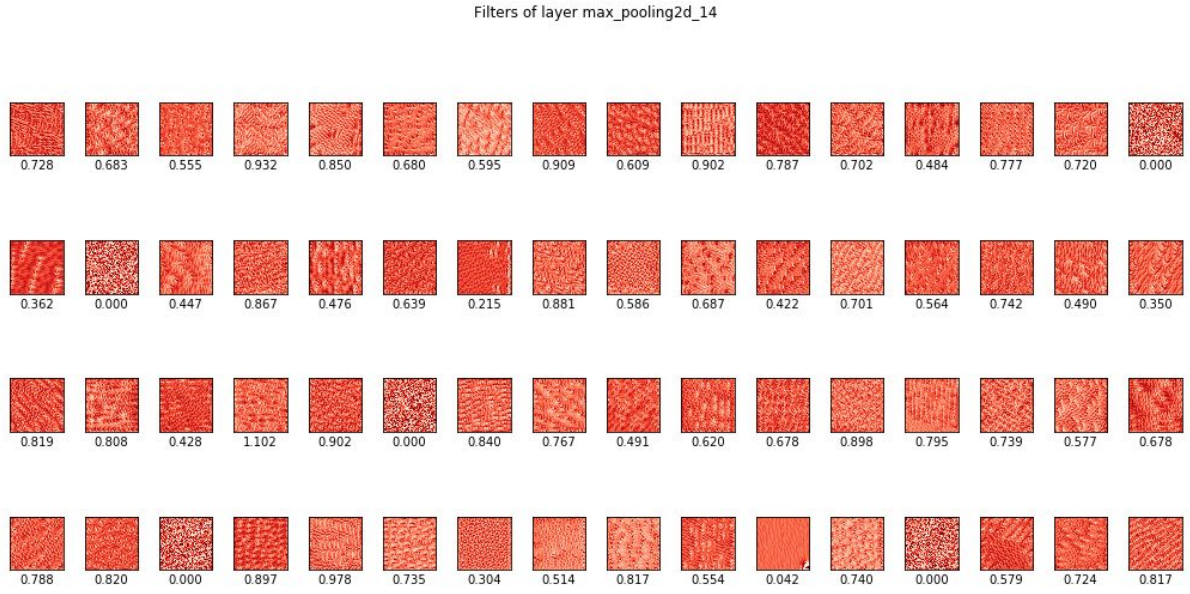


從 saliency maps可以發現，我的CNN主要注重再在觀察人臉的眼睛、眉毛、嘴巴部分，接著次要部分觀察人臉的輪廓，發現跟人類觀察的方法有所類似之處。

雖然覺得觀察的地方非常的符合道理、直覺，但其實我的CNN_model在這張圖片中並沒有預測正確，顯然在訓練這個model的時候還是有些缺失存在的。

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: r05229016 羅章碩)





以上兩張是我layer 0的輸入及layer max_polling2d_14之filters

發現有activate的參數蠻多的(可能是我在前面cov、polling layer並沒有加太多dropout)

所以挑>0.8的來講，首先最activate的神經元是第三排左邊數來第四個 1.102，他吃到的輸入是圖片左邊、下方的訊號，並且整張圖都有微弱訊號，

而第一行第一個(0.728)、第一行第八個(0.903)、第一行第十個(0.902)，在圖片靠近邊

緣的地方都有強烈的訊號，因此可以推論，這個filter大概是搜尋圖片邊框的訊號，特別是4個角(由第一行第八個(0.903)推知)。