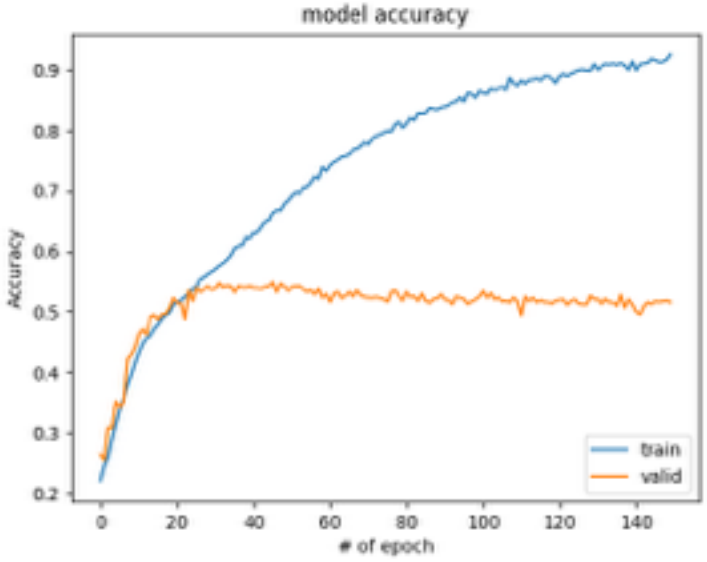
學號：R06921082 系級： 電機碩一 姓名：陳與賢

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？
答：

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 44, 44, 64)	1664
zero_padding2d_1 (ZeroPaddin	(None, 48, 48, 64)	0
max_pooling2d_1 (MaxPooling2	(None, 22, 22, 64)	0
zero_padding2d_2 (ZeroPaddin	(None, 24, 24, 64)	0
conv2d_2 (Conv2D)	(None, 22, 22, 64)	36928
zero_padding2d_3 (ZeroPaddin	(None, 24, 24, 64)	0
conv2d_3 (Conv2D)	(None, 22, 22, 64)	36928
average_pooling2d_1 (Average	(None, 10, 10, 64)	0
zero_padding2d_4 (ZeroPaddin	(None, 12, 12, 64)	0
conv2d_4 (Conv2D)	(None, 10, 10, 128)	73856
zero_padding2d_5 (ZeroPaddin	(None, 12, 12, 128)	0
conv2d_5 (Conv2D)	(None, 10, 10, 128)	147584
zero_padding2d_6 (ZeroPaddin	(None, 12, 12, 128)	0
average_pooling2d_2 (Average	(None, 5, 5, 128)	0
flatten_1 (Flatten)	(None, 3200)	0
dense_1 (Dense)	(None, 1024)	3277824
dropout_1 (Dropout)	(None, 1024)	0
dense_2 (Dense)	(None, 1024)	1049600
dropout_2 (Dropout)	(None, 1024)	0
dense_3 (Dense)	(None, 7)	7175

Total params: 4,631,559
Trainable params: 4,631,559
Non-trainable params: 0

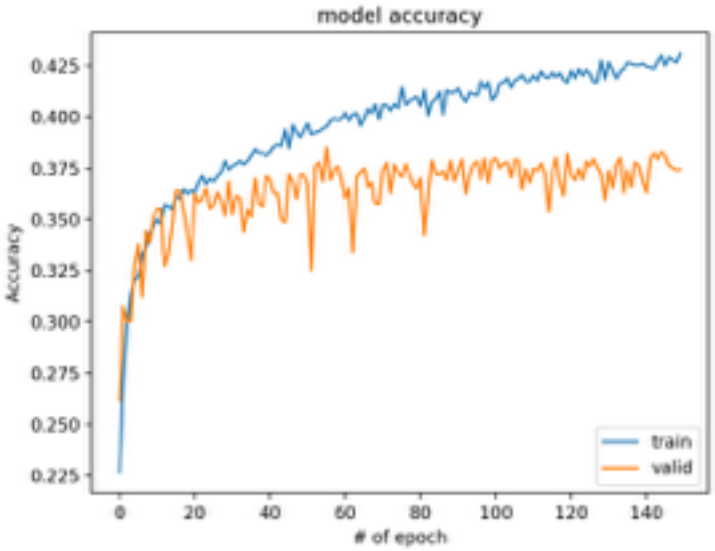


上圖為模型的架構（因為太長分成兩張），使用的model是助教提供的sample code的model，左圖是我的訓練過程，可以發現雖然在training data上最後的acc可以達到9成，但validation的acc並不高，所以在train到約第20個epoch的時候就已經overfitting了，後來我上傳到kaggle上的model有再額外使用keras的image generator來對input做preprocessing，就可以讓acc更高。

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？
答：

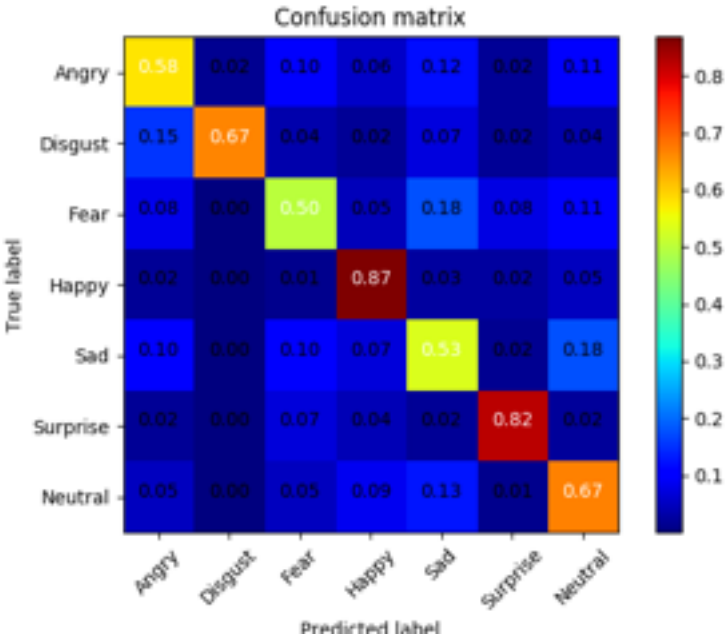
Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 48, 48, 1)	0
flatten_1 (Flatten)	(None, 2304)	0
dense_1 (Dense)	(None, 512)	1180160
dense_2 (Dense)	(None, 512)	262656
dense_3 (Dense)	(None, 512)	262656
dense_4 (Dense)	(None, 512)	262656
dense_5 (Dense)	(None, 1024)	525312
dense_6 (Dense)	(None, 1024)	1049600
dense_7 (Dense)	(None, 1024)	1049600
dense_8 (Dense)	(None, 7)	7175
activation_1 (Activation)	(None, 7)	0

Total params: 4,599,815
Trainable params: 4,599,815
Non-trainable params: 0



上圖是我的DNN model，湊出與CNN相同的參數量，訓練過程如左圖，可以發現acc比CNN還低而且震盪很明顯，由此可知對於影像的部分，CNN是可以獲得一些好處的，另外可以發現的是DNN與CNN都是在epoch約20的時候acc就開始收斂了。

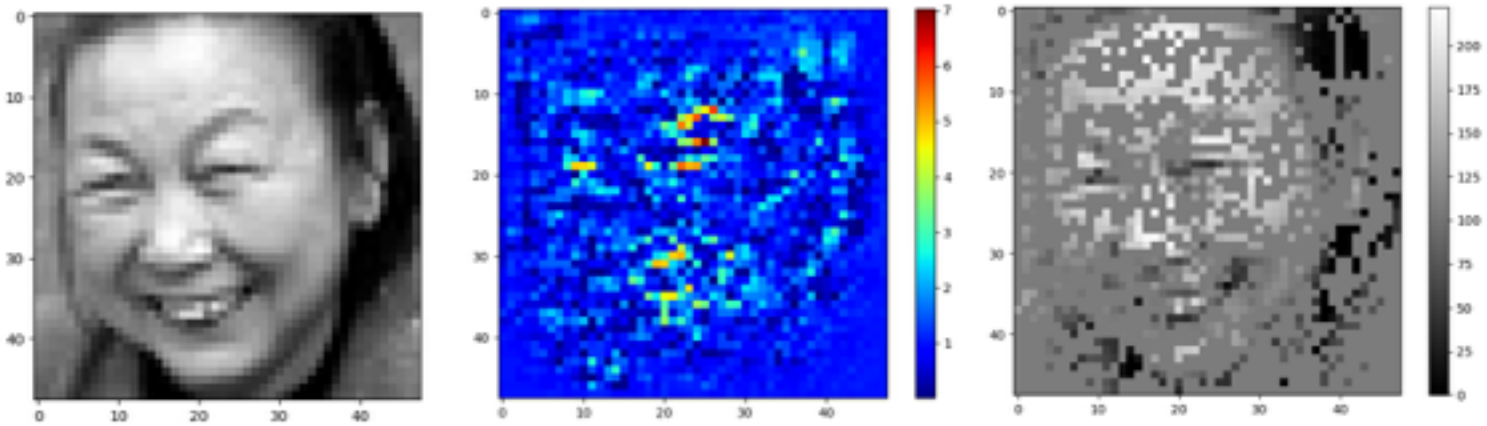
3. (1%) 觀察答錯的圖片中，哪些 class 彼此間容易用混？[繪出 confusion matrix 分析]
答：



由左圖的confusion matrix可以發現，Angry有超過10%的機率被認成Sad及Neutral、Disgust有15%的機率被認成Angry、Fear有將近20%被認成Sad、Sad也有將近20%被認成Neutral、Neutral則有約13%被認成Sad，而Happy跟Surprise則有蠻高的命中率。

這其實可以推理出來，有些人生氣時是面無表情的、厭惡到極致時也會感到憤怒、十分恐懼時的表情與傷心的表情很相近...等等。

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？
答：



左圖為輸入，中間為saliency map，右圖為mask後的結果，由此可知主要是focus在人的嘴巴以及眉毛，我想應該是因為人的各種表情在嘴巴與眉毛的弧度會有明顯的變化，才可以易於判定是什麼表情（此種方法也與人類在判斷別人的情緒相同）。

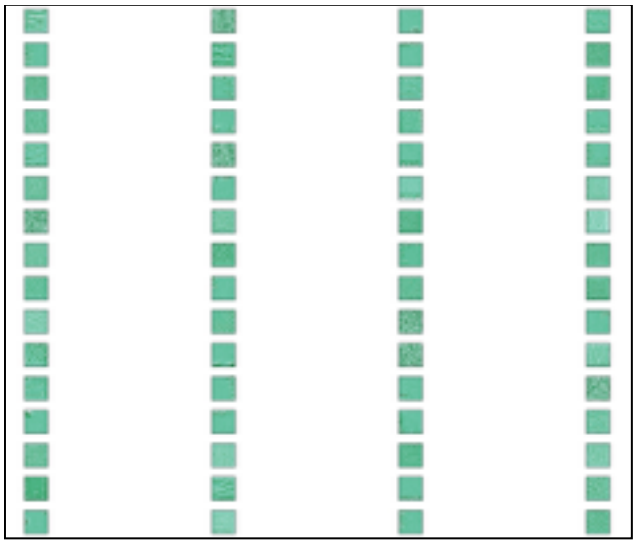
5. (1%) 承(1)(2)，利用上課所提到的 gradient ascent 方法，觀察特定層的filter最容易被哪種圖片activate。
答：
輸入同第4題

layer0:

layer1:

layer2:

filter:



可以發現在愈後面的layer五官會愈來愈明顯，而且偵測表情也應當是用五官來判定，所以五官明顯及表情誇張的人愈容易被active。