學號：B03502125 系級： 機械四 姓名：倪嘉宏

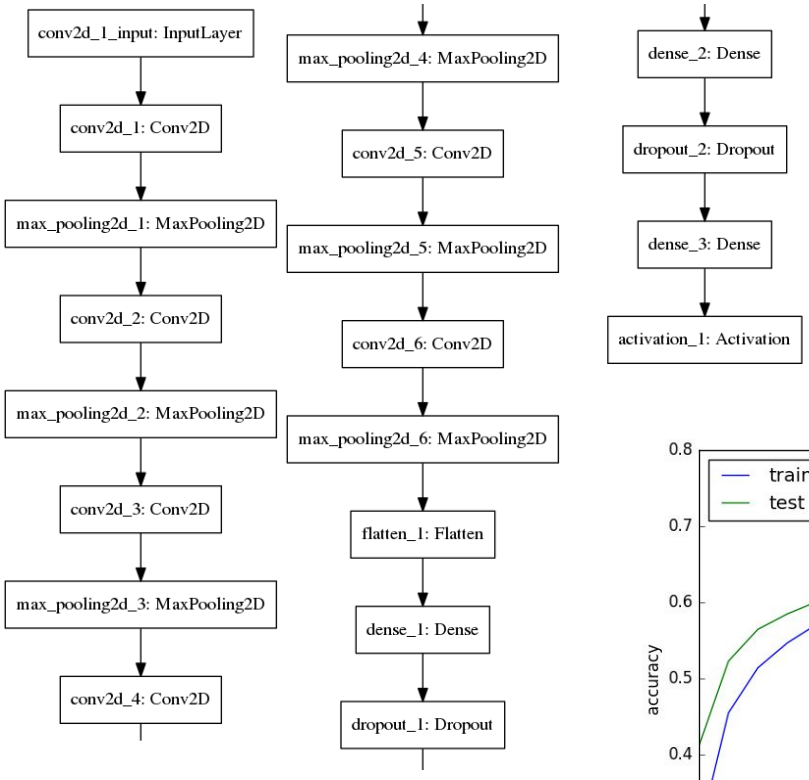
1. (1%) 請說明你實作的 CNN model，其模型架構、訓練過程和準確率為何？
   (Collaborators:)

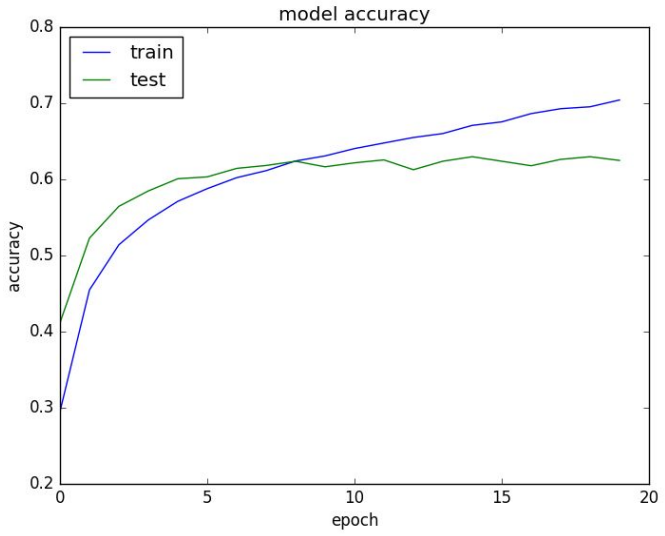
   答：Total parameter : 2230151

CNN的架構參考許多前人經驗架設而成，主要是將輸入的資料反覆做convolution2D以及maxPooling2D（曾經也嘗試過混入averagePooling2D和batchNormaliztion等等層，但結果並沒有太卓越的提升）之後接flatten layer再接數個dense layer。dense layer之間可以適當的加上幾個dropout layer，會對overfitting的狀況有所改善，但是加太多又會train不動因此適量就好。

訓練過程可以看到training data的accuracy在第七個epoch之後超過60，並仍然持續上升。但是validation accuracy卻在抵達0.62之後再0.02的區間跳動，不再有顯著的上升。藉由增加dropout層可以改善validation accuracy背training accuracy狠狠甩掉的狀況，但會減慢訓練的速度。

最終準確率這樣跑出來的結果經過反覆調校之後可以達到0.64左右。但是要上傳kaggle上用來衝過strong baseline的結果要以數個差異較大的model進行ensemble，可略高於0.67。



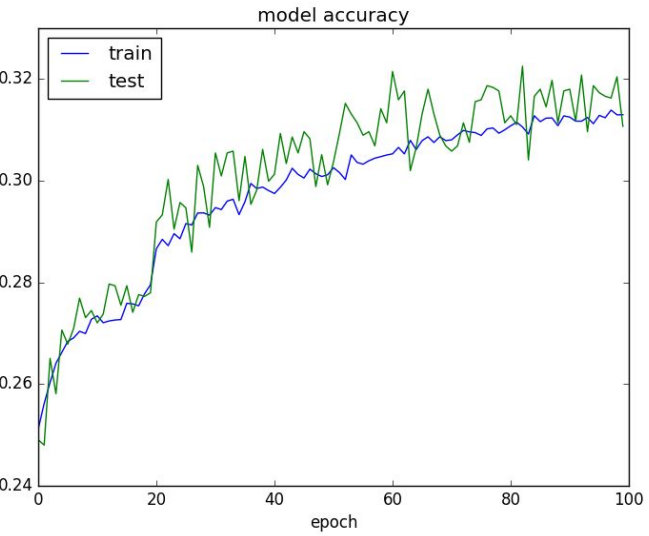


2. (1%) 承上題，請用與上述 CNN 接近的參數量，實做簡單的 DNN model。其模型架構、訓練過程和準確率為何？試與上題結果做比較，並說明你觀察到了什麼？
(Collaborators:)

答：Total parameters: 814991

使用DNN時，參數數量增加會讓GPU的記憶體不堪負荷。在這裡我的GPU還能負荷的前提下儘量增加參數數量至接近第一題的數量製作了這個model。儘管參數數量增加會使GPU記憶體不堪負荷，但是整體的運算時間卻相較於CNN的情況有著顯著的提升。

另外值得一提的是，雖然每個epoch所耗費的時間減少，但是所需要達到穩定的準確律所需要的epoch數量也跟著增加，因此整體所耗費的訓練時間並沒有節省多少，而且最終準確度並不理想。

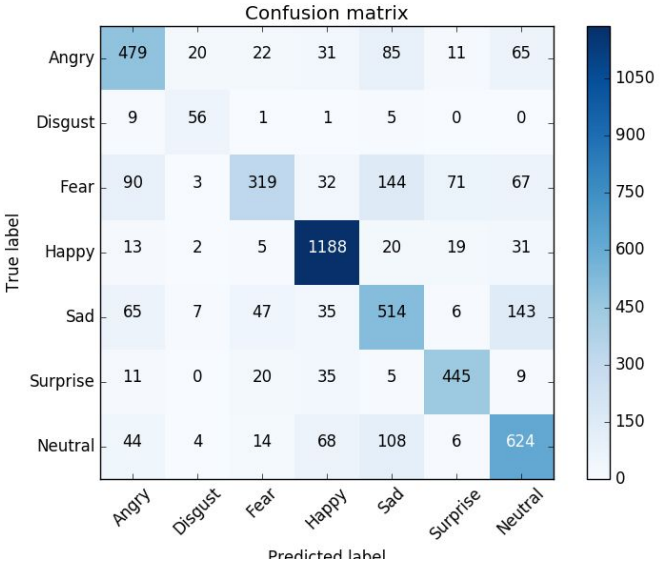


Layer (type)	Output Shape	Param #
max_pooling2d_1 (MaxPooling2	(None, 24, 24, 1)	0
dense_1 (Dense)	(None, 24, 24, 64)	128
max_pooling2d_2 (MaxPooling2	(None, 12, 12, 64)	0
dense_2 (Dense)	(None, 12, 12, 64)	4160
max_pooling2d_3 (MaxPooling2	(None, 6, 6, 64)	0
dense_3 (Dense)	(None, 6, 6, 64)	4160
max_pooling2d_4 (MaxPooling2	(None, 3, 3, 64)	0
dense_4 (Dense)	(None, 3, 3, 64)	4160
max_pooling2d_5 (MaxPooling2	(None, 2, 2, 64)	0
dense_5 (Dense)	(None, 2, 2, 64)	4160
flatten_1 (Flatten)	(None, 256)	0
dense_6 (Dense)	(None, 8)	2056
dense_7 (Dense)	(None, 512)	4608
dense_8 (Dense)	(None, 512)	262656
dense_9 (Dense)	(None, 512)	262656
dense_10 (Dense)	(None, 512)	262656
dense_11 (Dense)	(None, 7)	3591
activation_1 (Activation)	(None, 7)	0

Total params: 814,991
Trainable params: 814,991
Non-trainable params: 0

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：右圖為混淆矩陣，以5000筆資料製作。



從繪製出來的confusion matrix可以看出不同分類被錯誤歸類的情形。觀察對角線以外數字較大的那幾個區塊即可知道哪些class彼此之間容易混淆。例如最大的數字144，為誤將Fear認成Sad的數量。而次大的143為誤將Sad辨認成Neutral的數量。第三大的108惟物將Neutral辨認成Sad的數量。

如果將左上右下的數字加起來（即表格對角線對折），可得到某兩種表情混淆的數量。依此可以看出程式最容易將Sad和Neutral互相混淆。

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:)

   答：

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:)

   答：