

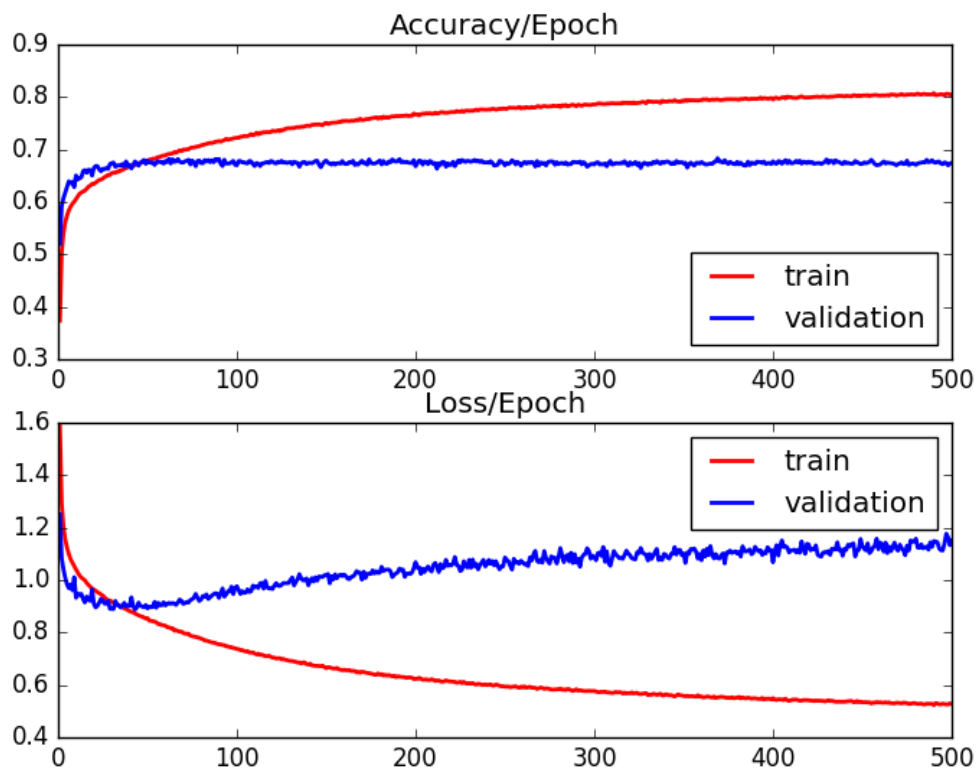
ML HW3 Report

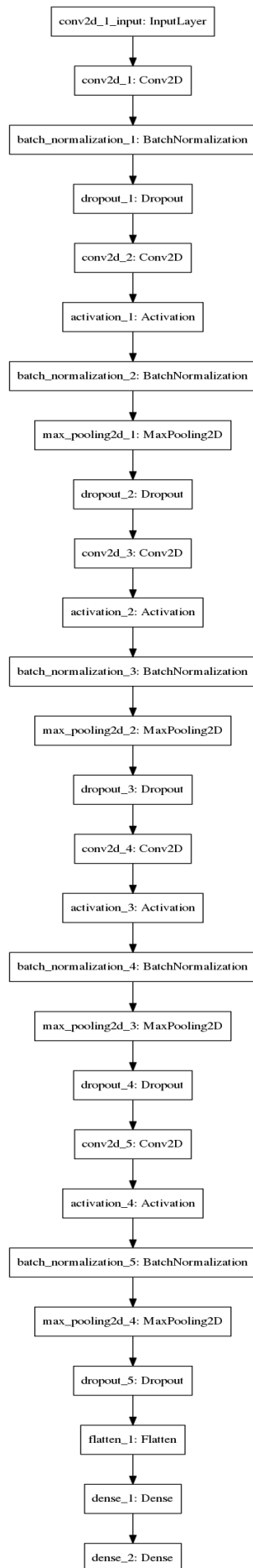
學號：B04611015 系級：資工二 姓名：陳佳佑

Q1. (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？

附圖為我的 model 架構，原先一開始採用教授的三層的 CNN 加 Dense 還有 dropout，但是效果不太好，經過調整參數後，只有剛好過 baseline。之後就開始加 CNN Layer 層數，四層時的上限大概是 60~61%，五層時為 63~64%。原先有想過用 histogram equalization 做亮度平衡，但是看了幾張圖後，覺得沒有必要，因為抽樣的結果是亮度都還可以。於是，採用了增加 training data 的策略，將原先的圖片水平翻轉，並且上下左右 shift 0.2，製造更多資料，而最佳的結果也到 68.8%(但是忘記存 model)，平均的結果在 68%上下。

後 6709 筆為 validation set，Training 之結果，可以看到準確度一直在 67~68%間徘徊，而 validation 的結果卻在上升，代表後面有些微的 overfitting。最後，我將 validation data 丟進去，然後 Train 120 個 epoch 得到 kaggle best 69.016%。原先還想將所有的 model 拿出來做 ensemble，但是看到 Deadline 快到了，所以沒做。



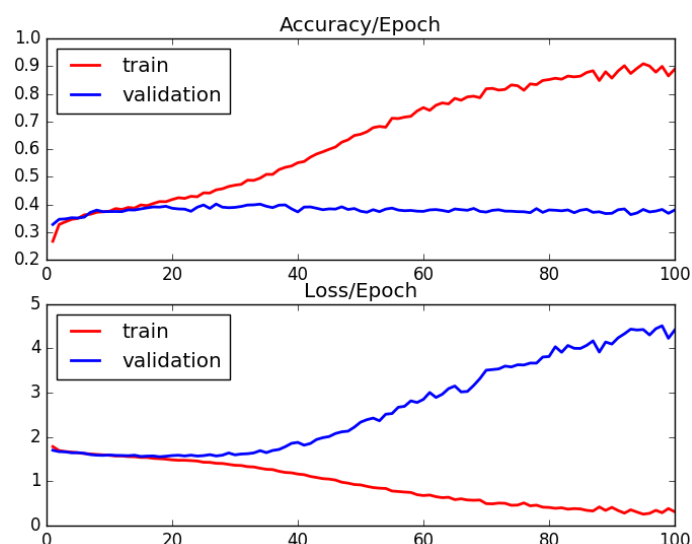


Layer (type)	Output Shape	Param #
=====		
conv2d_1 (Conv2D)	(None, 46, 46, 25)	250
batch_normalization_1 (Batch Normalization)	(None, 46, 46, 25)	100
dropout_1 (Dropout)	(None, 46, 46, 25)	0
conv2d_2 (Conv2D)	(None, 44, 44, 50)	11300
activation_1 (Activation)	(None, 44, 44, 50)	0
batch_normalization_2 (Batch Normalization)	(None, 44, 44, 50)	200
max_pooling2d_1 (MaxPooling2D)	(None, 22, 22, 50)	0
dropout_2 (Dropout)	(None, 22, 22, 50)	0
conv2d_3 (Conv2D)	(None, 20, 20, 100)	45100
activation_2 (Activation)	(None, 20, 20, 100)	0
batch_normalization_3 (Batch Normalization)	(None, 20, 20, 100)	400
max_pooling2d_2 (MaxPooling2D)	(None, 10, 10, 100)	0
dropout_3 (Dropout)	(None, 10, 10, 100)	0
conv2d_4 (Conv2D)	(None, 8, 8, 125)	112625
activation_3 (Activation)	(None, 8, 8, 125)	0
batch_normalization_4 (Batch Normalization)	(None, 8, 8, 125)	500
max_pooling2d_3 (MaxPooling2D)	(None, 4, 4, 125)	0
dropout_4 (Dropout)	(None, 4, 4, 125)	0
conv2d_5 (Conv2D)	(None, 2, 2, 250)	281500
activation_4 (Activation)	(None, 2, 2, 250)	0
batch_normalization_5 (Batch Normalization)	(None, 2, 2, 250)	1000
max_pooling2d_4 (MaxPooling2D)	(None, 1, 1, 250)	0
dropout_5 (Dropout)	(None, 1, 1, 250)	0
flatten_1 (Flatten)	(None, 250)	0
dense_1 (Dense)	(None, 1000)	251000
dense_2 (Dense)	(None, 7)	7007
=====		
Total params: 710,982		
Trainable params: 709,882		
Non-trainable params: 1,100		

Q2. (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？

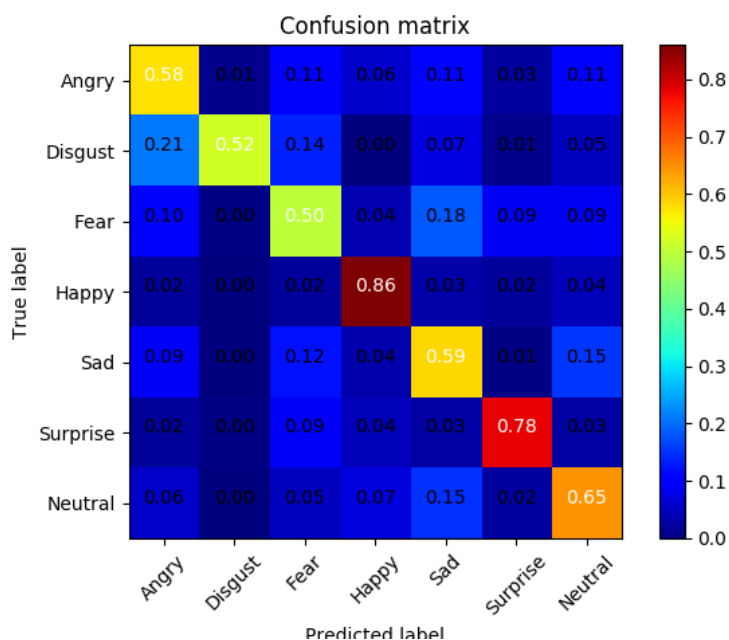
我的 model 用了五層的 dense(relu) 跟一層 dense(softmax)，最後在 validation set 上的準確率只有到 39% 左右。很明顯的，DNN 在做影像辨識上比 CNN 難很多，因為 CNN 有用 Max pooling 和 Convolution 這些有利用到影像處理特性的技巧，而 DNN 又全都是 Fully connected，很容易 Overfitting，而我的模型很明顯的也在 20 個 epoch 左右後就開始 overfitting。除此之外，與 CNN 相比，DNN 的 Dropout 比例非常不好抓，原先我想要透過 DropOut 抑制 Overfitting，卻發現僅微調幾個參數，整個 Model 就 Underfitting，完全無法起飛。

Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 256)	590080
dense_2 (Dense)	(None, 256)	65792
dense_3 (Dense)	(None, 256)	65792
dense_4 (Dense)	(None, 256)	65792
dense_5 (Dense)	(None, 256)	65792
dense_6 (Dense)	(None, 7)	1799
Total params: 855,047		
Trainable params: 855,047		
Non-trainable params: 0		



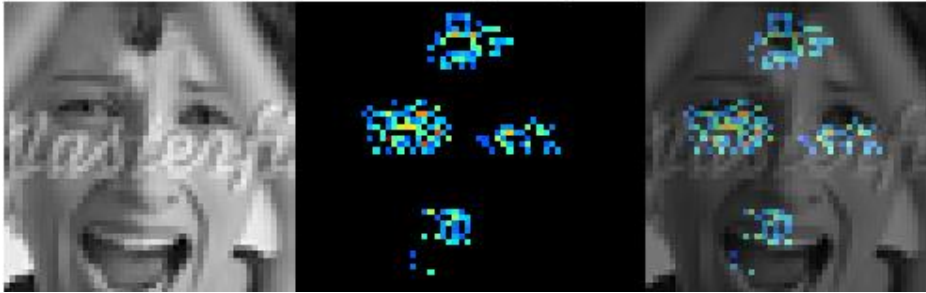
Q3. (1%) 觀察答錯的圖片中，哪些 class 彼此間容易用混？[繪出 confusion matrix 分析]

在我的 model 中，(Angry, Neutral), (Fear, Sad), (Disgust, Angry), (Disgust, Fear)，很容易被誤判。



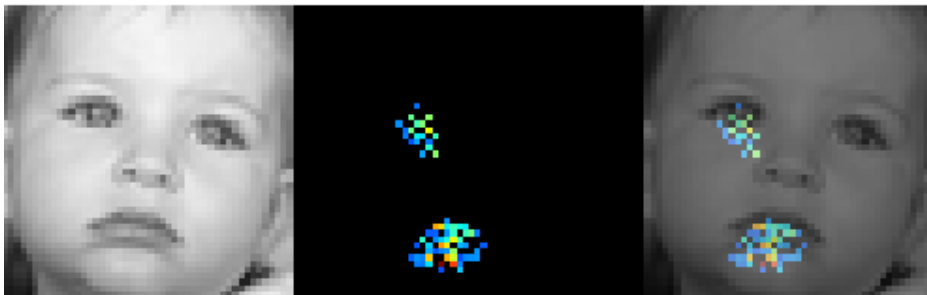
Q4. (1%) 從(1)(2)可以發現，使用 CNN 的確有些好處，試繪出其 saliency maps，觀察模型在做 classification 時，是 focus 在圖片的哪些部份？

Saliency map



在這個例子上，模型在座 classification 時 focus 在眼睛與嘴巴上緣，而最上面感應強烈的部分是對雜訊產生反應。

Saliency map



在這張圖上，右眼與嘴巴都有強烈的反應。

Saliency map



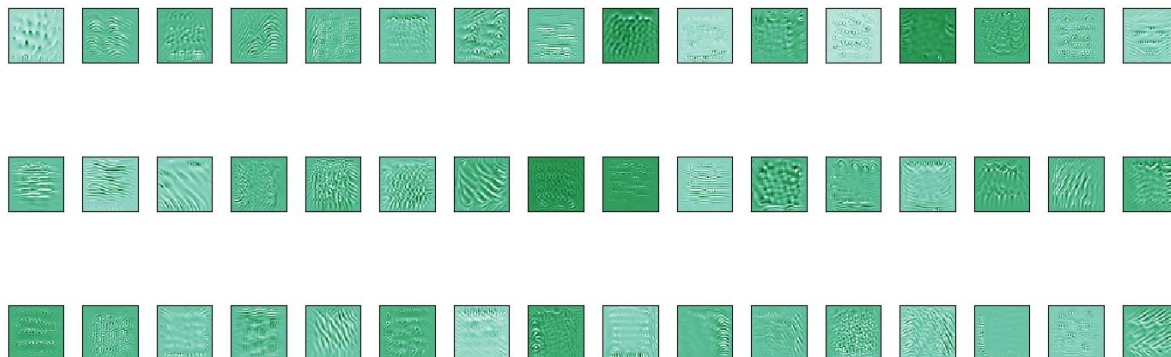
在這張圖上，嘴巴上緣有強烈反應。

根據上面三張圖，我們可以知道，在判斷表情時，嘴巴上緣是一個很重要的特徵，不論在哪張圖中，嘴巴上緣都有強烈的反應。

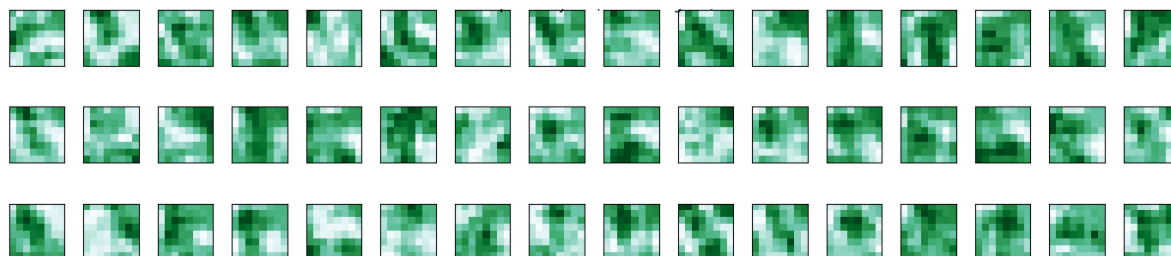
Q5. (1%) 承(1)(2)，利用上課所提到的 **gradient ascent** 方法，觀察特定層的 **filter** 最容易被哪種圖片 **activate**。

根據下圖，可以觀察到第四層的 **layer** 主要被拿來判斷波紋，而我在實作時發現前兩層都是雜訊，後來仔細檢查 **model**，發現是 **Dropout** 設的過小，造成只有後兩層有較明顯的波紋。

Conv2d_4 detect



Conv2d_4 image



[Bonus 1] 從 **training data** 中移除部份 **label**，實做 **semi-supervised learning**

我以 **kaggle best** 但是有切 **validation set** 的 **model** 做為 **supervised learning** 的結果，然後再依照利用 **model predict testing set** 後，把 **predict** 的結果與 **training data** 和在一起 **train**。與第一題的結果相比較，我們可以觀察到 **self-training overfitting** 的速度比原先的 **CNN** 還快，因為 **self-training model** 在 **training** 時的 **data**，有部分是以上一個 **model** 為基礎的結果，因此更容易 **overfitting**，但是對於準確率的提升是有幫助的，在第一題的準確率是 67~68%而 **self-training** 丟到 **kaggle** 上後的準確率是 68.8%，如果再把 **validation data** 丟入，應該可以再提升一些。

