學號：R06944051 系級： 網媒碩一 姓名：郭柏辰

1. (1%) 請說明你實作的 CNN model，其模型架構、訓練參數和準確率為何？

答：

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 48, 48, 32)	320
conv2d_2 (Conv2D)	(None, 48, 48, 32)	9248
batch_normalization_1 (Batch	(None, 48, 48, 32)	128
max_pooling2d_1 (MaxPooling2	(None, 24, 24, 32)	0
dropout_1 (Dropout)	(None, 24, 24, 32)	0
conv2d_3 (Conv2D)	(None, 24, 24, 64)	18496
conv2d_4 (Conv2D)	(None, 24, 24, 64)	36928
batch_normalization_2 (Batch	(None, 24, 24, 64)	256
max_pooling2d_2 (MaxPooling2	(None, 12, 12, 64)	0
dropout_2 (Dropout)	(None, 12, 12, 64)	0
conv2d_5 (Conv2D)	(None, 12, 12, 128)	73856
conv2d_6 (Conv2D)	(None, 12, 12, 128)	147584
batch_normalization_3 (Batch	(None, 12, 12, 128)	512
max_pooling2d_3 (MaxPooling2	(None, 6, 6, 128)	0
dropout_3 (Dropout)	(None, 6, 6, 128)	0
conv2d_7 (Conv2D)	(None, 6, 6, 256)	295168
conv2d_8 (Conv2D)	(None, 6, 6, 256)	590080
batch_normalization_4 (Batch	(None, 6, 6, 256)	1024
max_pooling2d_4 (MaxPooling2	(None, 3, 3, 256)	0
dropout_4 (Dropout)	(None, 3, 3, 256)	0
flatten_1 (Flatten)	(None, 2304)	0
dense_1 (Dense)	(None, 256)	590080
batch_normalization_5 (Batch	(None, 256)	1024
dropout_5 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 512)	131584
batch_normalization_6 (Batch	(None, 512)	2048
dropout_6 (Dropout)	(None, 512)	0
dense_3 (Dense)	(None, 7)	3591

Model: 我執行 conv2d => conv2d=> batch_normalization=>max_pooling2d=>dropout 四次，之後再 Flatten 接上兩層 Dense=> batch_normalization=>dropout，最後再 output。
參數設定如下:
conv2d: kernel size 為 3x3，padding: same，filter 個數 32,32,64,64,128,128,256,256。
max_pooling2d: pool_size=(2, 2), padding='same'。
dropout: 分別為 0.2，0.25，0.4，0.4，0.5，0.5。
Dense: 第一層為 256，第二層為 512。
準確率: Public: 0.69350 Private: 0.67957。

2. (1%) 請嘗試 data normalization, data augmentation,說明實行方法並且說明對準確率有什麼樣的影響？

答：

實作前準確率為: Public: 0.66174 Private: 0.64641。
data normalization: 我針對 pixel 的位置，個別去算整個 dataset 該位置的平均值及標準差。
實作後準確率: Public: 0.65115 Private: 0.64530。

我實作 data normalization 後發現準確率好像差不多，可能是因為我在實作前有先對整張圖/255，我想這可能也算是 normalization 的一種，所以結果差不多。

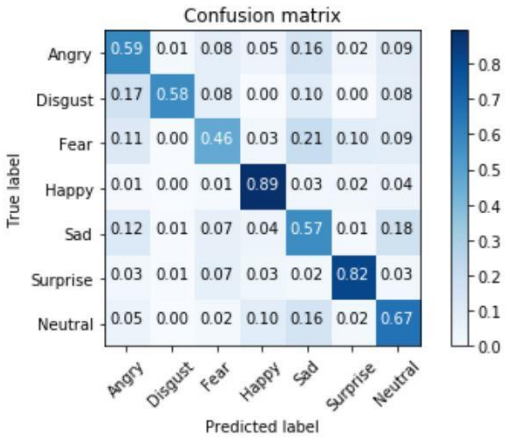
實作前準確率為: Public: 0.66174 Private: 0.64641。
data augmentation: 我使用 keras 的 ImageDataGenerator 做預先處理，對圖片做隨機旋轉、隨機平移、反轉，參數如下。

```
#Image PreProcessing
gen_img = ImageDataGenerator(rotation_range=20,
                             width_shift_range=0.1,
                             height_shift_range=0.1,
                             horizontal_flip=True)
```

實作後準確率: Public: 0.68041 Private: 0.67985。
我實作 data augmentation 後準確率明顯有提升，因為 data augmentation 可以製造一些假資料來增加訓練資料，訓練資料變多了所以訓練的效果也比較好。

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]

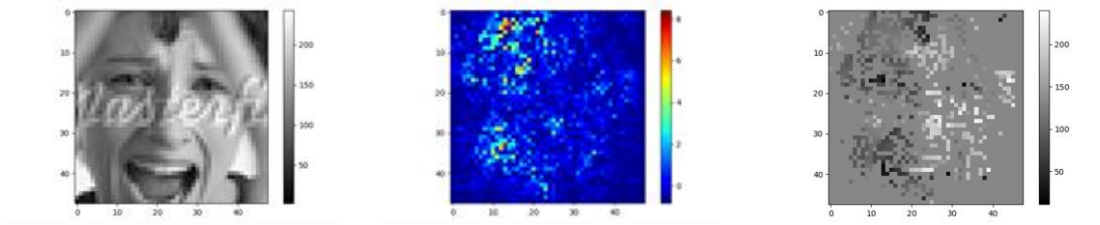
   答：

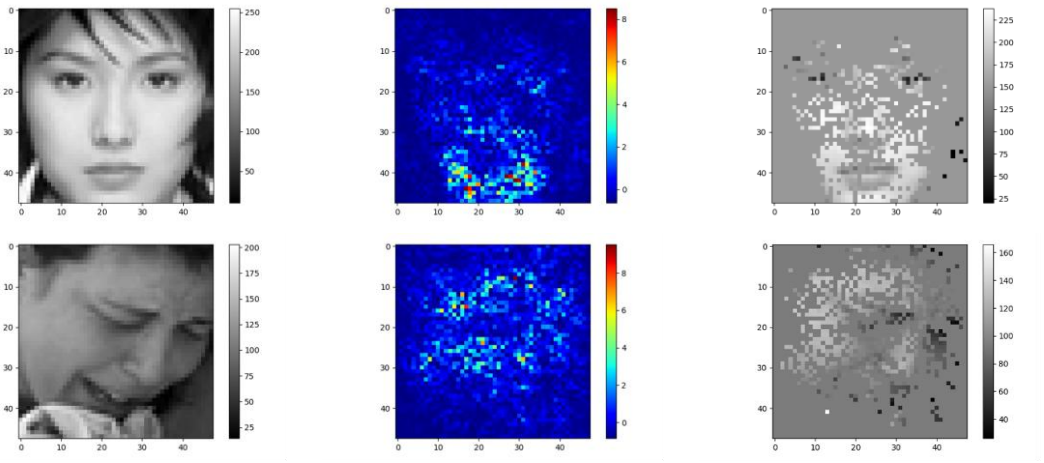


   在我訓練的模型中，可以發現生氣容易被誤判成噁心，傷心容易被誤判成害怕，中立容易被誤判成傷心。

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：

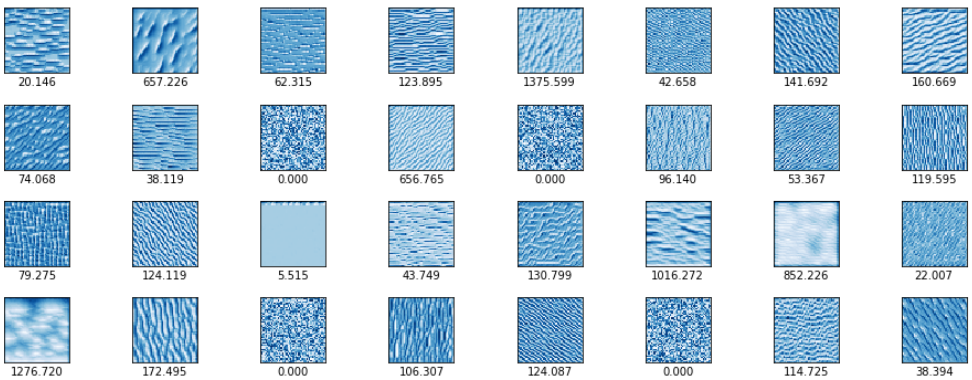




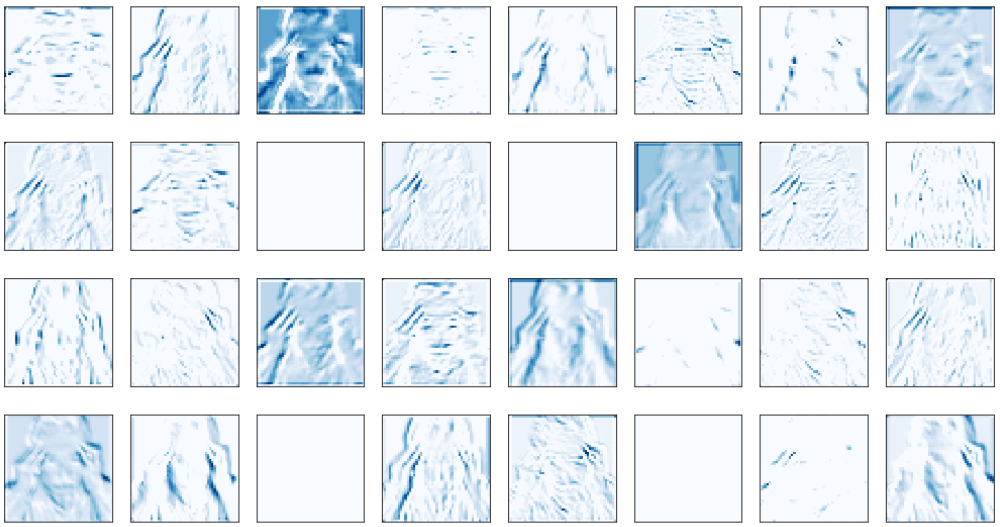
可以發現大部分都 focus 在眼睛跟嘴巴的部分，這跟我們人類判斷情緒時會觀察的特徵差不多。

5. (1%) 承(4) 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 與觀察 filter 的 output。

答：



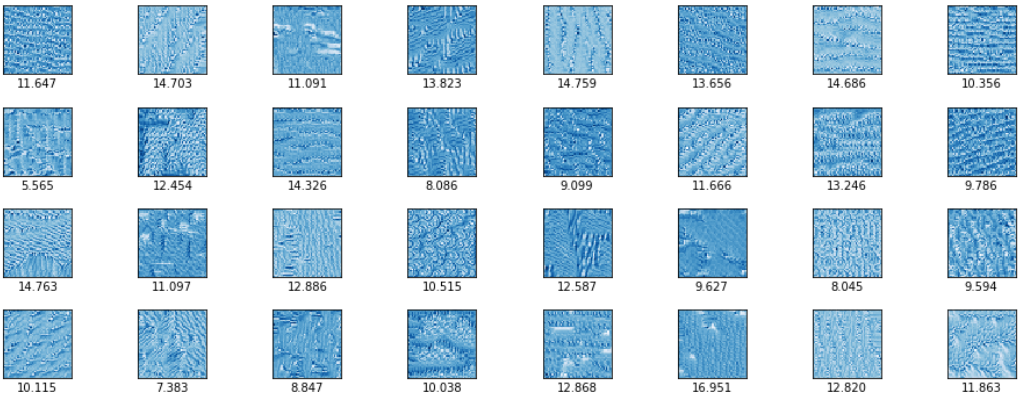
conv2d 第 2 層 filter



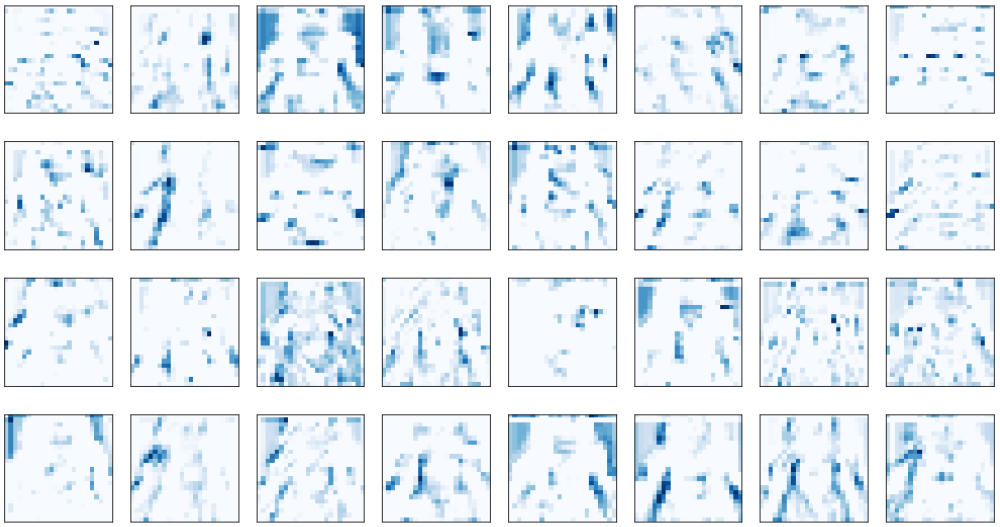
conv2d 第 2 層 output

顯示 conv2d 第二層 32 個 filter，由輸出的圖片可以發現大部分的圖片都只有臉的輪廓，這表

示可以找到人臉的位置。



conv2d 第 3 層 filter



conv2d 第 3 層 output

隨著層數越深，output 越無法辨識，我想這可能跟 max_pooling 把圖縮小有關。