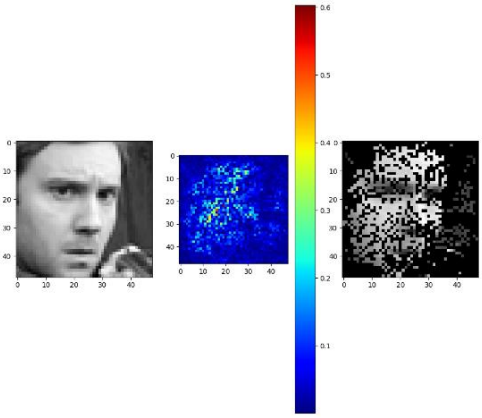
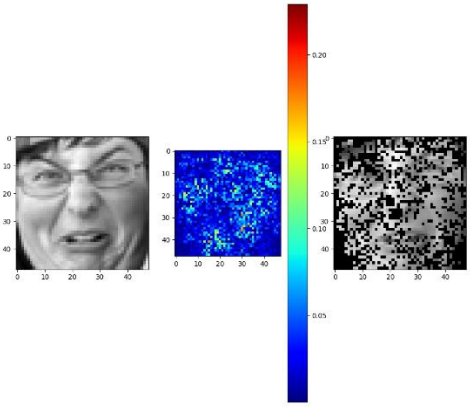
學號：R07943095 系級：EDA 碩一 姓名：劉世棠

1. (2%) 從作業三可以發現，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: 劉治硯、吳辰鋐)

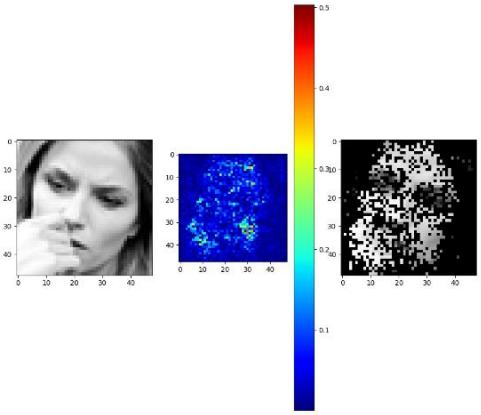
   以下依序為七種表情(Angry,Disgust,Fear,Happy,Sad,Surprise,Neutral):



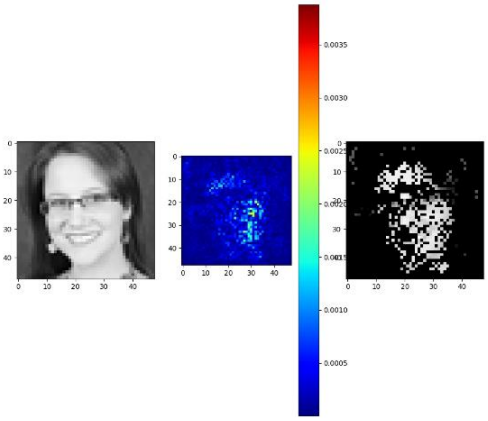
3-1 Angry



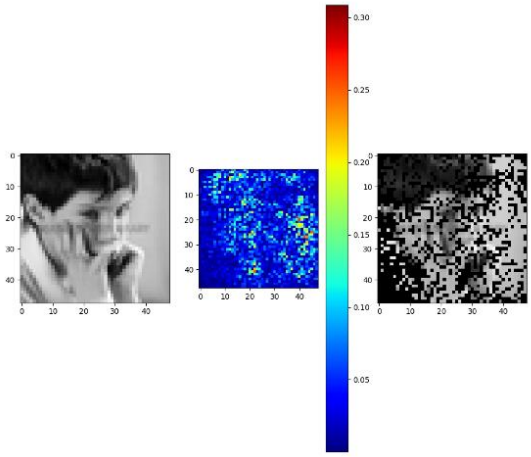
3-2 Disgust



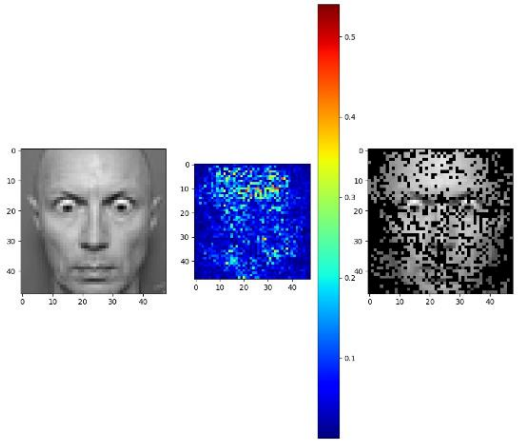
3-3 Fear



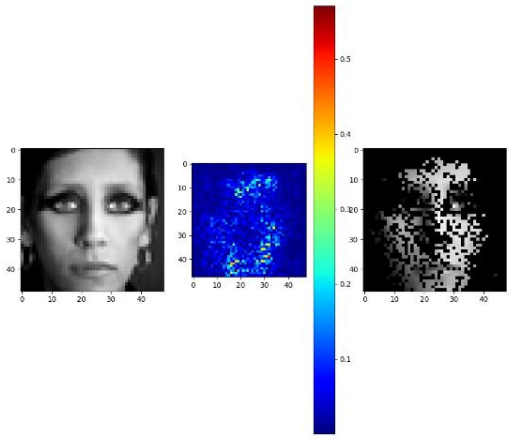
3-4 Happy



3-5 Sad



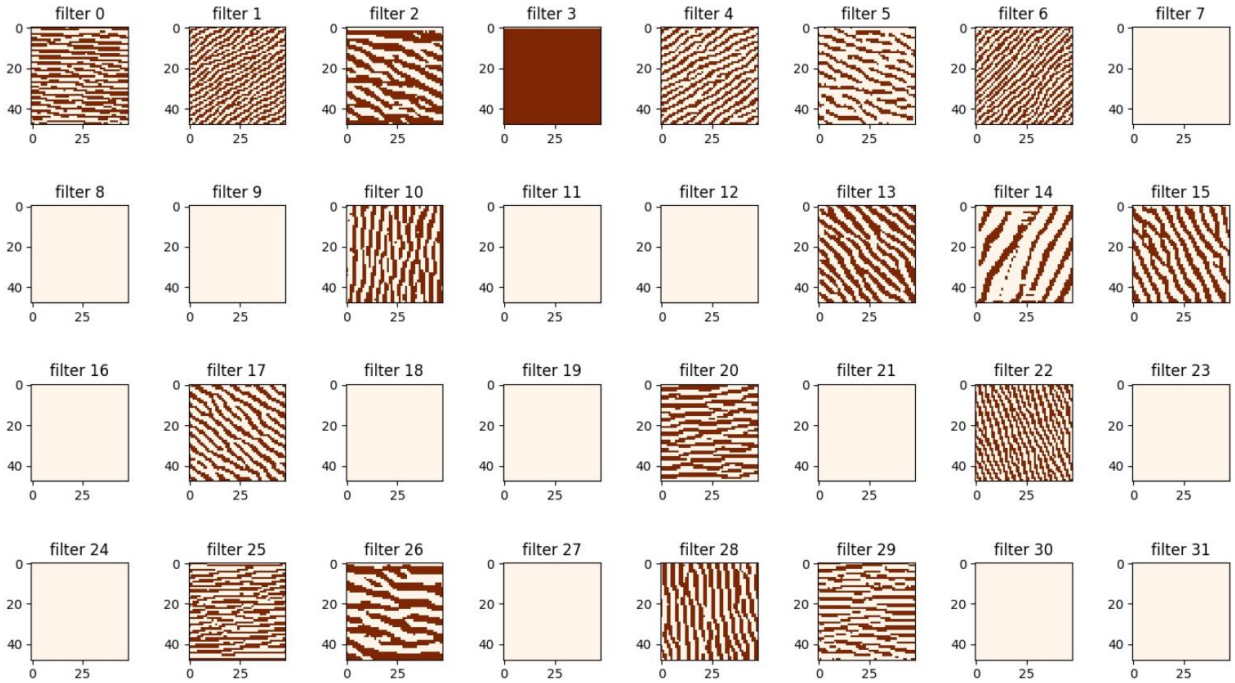
3-6 Surprise

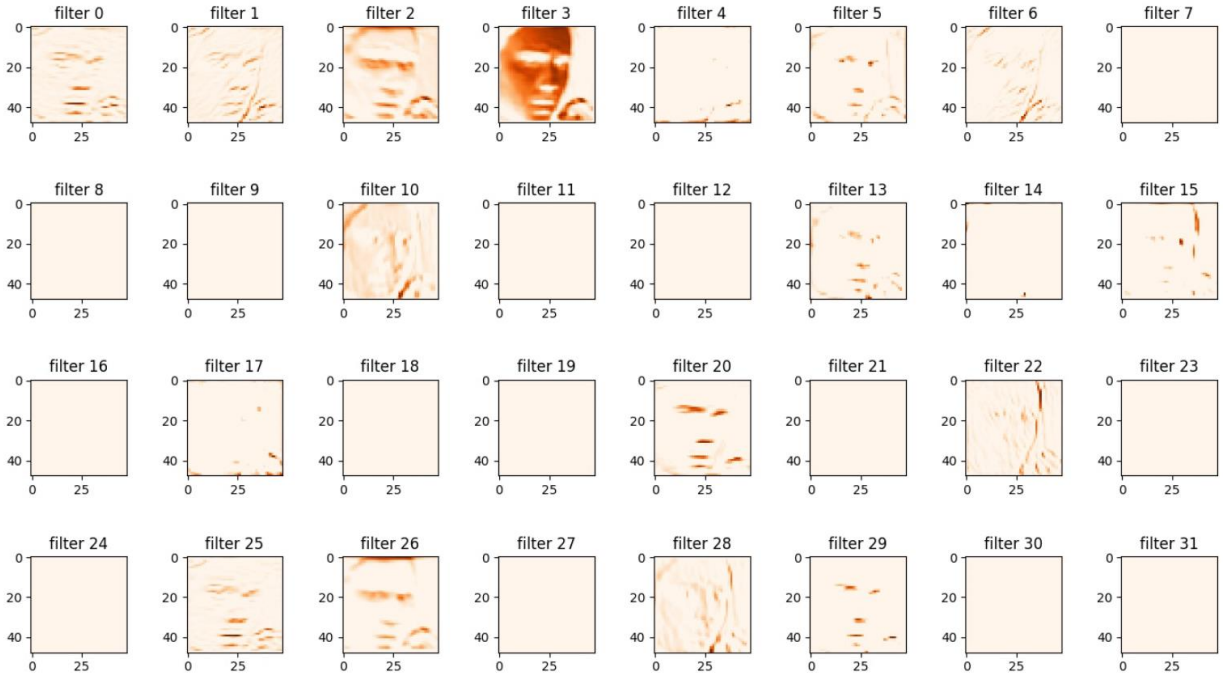


3-7 Neutral

⇨ 用 saliency map 當作 mask 後可以發現留下來的都是臉的部分，且對額頭與臉頰都比較有反應，這暗示我的 model 應該是真的有學到些什麼。

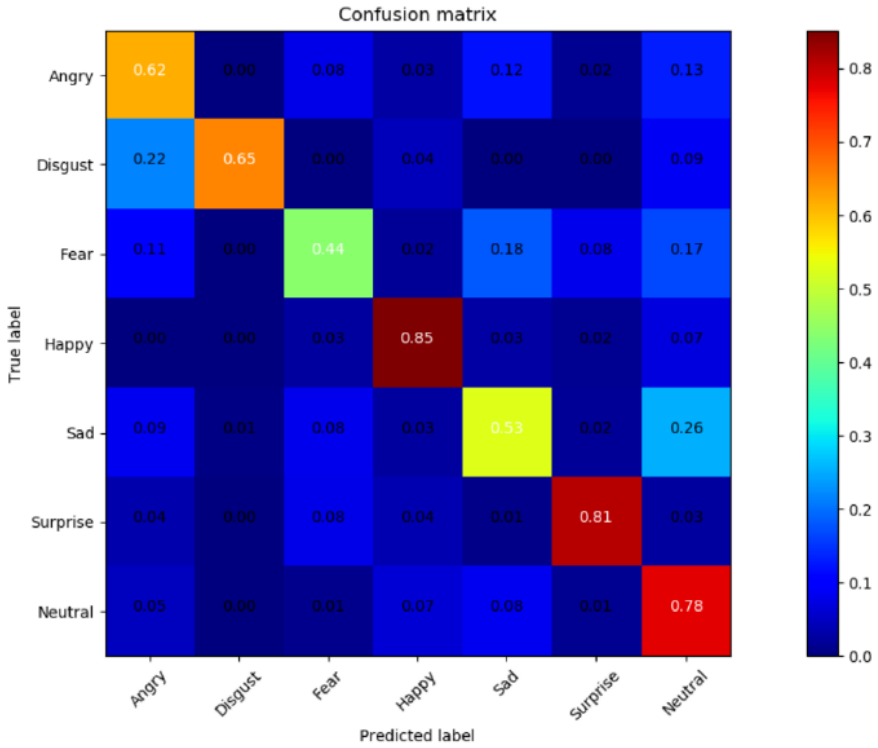
2. (3%) 承(1) 利用上課所提到的 gradient ascent 方法，觀察特定層的 filter 最容易被哪種圖片 activate 與觀察 filter 的 output。(Collaborators: 劉治硯、吳辰鋐)



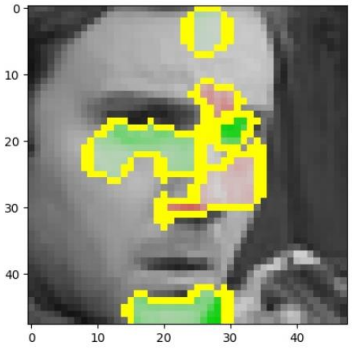


⇨ 兩張圖片是取第一層的前 32 個 filter，可以發現確實會注意臉部的輪廓，因此會有辦法辨識表情，此外權重太低(顏色淡)的 filter 則效果不好。

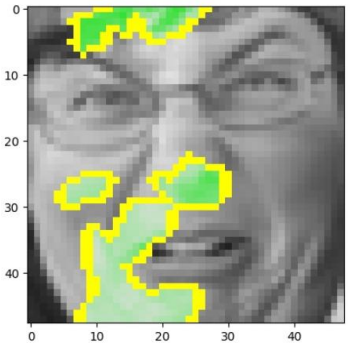
3. (3%) 請使用 Lime 套件分析你的模型對於各種表情的判斷方式，並解釋為何你的模型在某些 label 表現得特別好 (可以搭配作業三的 Confusion Matrix)。



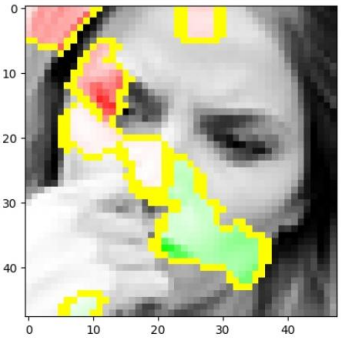
以下依序為七種表情(Angry,Disgust,Fear,Happy,Sad,Surprise,Neutral):



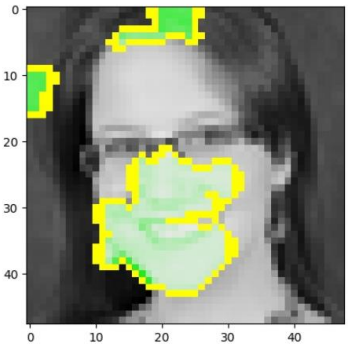
3-1 Angry



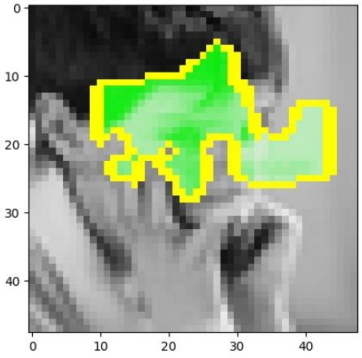
3-2 Disgust



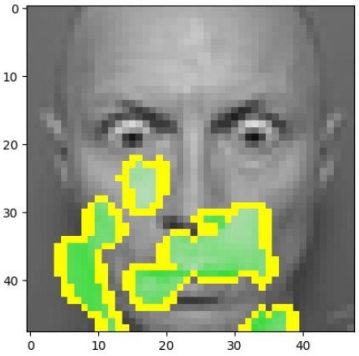
3-3 Fear



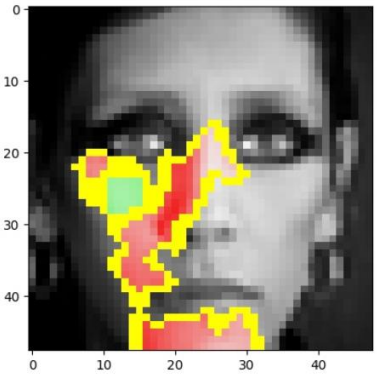
3-4 Happy



3-5 Sad



3-6 Surprise



3-7 Neutral

⇨ 我的模型在 Happy 的時候有較好的準確率，而以 Lime 套件來觀察，可以發現他會注意在嘴巴的部分，而人在開心時通常會笑出來，因此才會有這樣的結果。

4. (2%) [自由發揮] 請同學自行搜尋或參考上課曾提及的內容，實作任一種方式來觀察 CNN 模型的訓練，並說明你的實作方法及呈現 visualization 的結果。

   a. 利用 summary()

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 48, 48, 64)	1664
batch_normalization_1 (Batch	(None, 48, 48, 64)	256
max_pooling2d_1 (MaxPooling2	(None, 24, 24, 64)	0
dropout_1 (Dropout)	(None, 24, 24, 64)	0
conv2d_2 (Conv2D)	(None, 24, 24, 128)	73856
batch_normalization_2 (Batch	(None, 24, 24, 128)	512
max_pooling2d_2 (MaxPooling2	(None, 12, 12, 128)	0
dropout_2 (Dropout)	(None, 12, 12, 128)	0
conv2d_3 (Conv2D)	(None, 12, 12, 256)	295168
batch_normalization_3 (Batch	(None, 12, 12, 256)	1024
max_pooling2d_3 (MaxPooling2	(None, 6, 6, 256)	0
dropout_3 (Dropout)	(None, 6, 6, 256)	0
conv2d_4 (Conv2D)	(None, 6, 6, 512)	1180160
batch_normalization_4 (Batch	(None, 6, 6, 512)	2048
max_pooling2d_4 (MaxPooling2	(None, 3, 3, 512)	0
dropout_4 (Dropout)	(None, 3, 3, 512)	0
flatten_1 (Flatten)	(None, 4608)	0
dense_1 (Dense)	(None, 512)	2359808
activation_1 (Activation)	(None, 512)	0
batch_normalization_5 (Batch	(None, 512)	2048
dropout_5 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 512)	262656
activation_2 (Activation)	(None, 512)	0
batch_normalization_6 (Batch	(None, 512)	2048
dropout_6 (Dropout)	(None, 512)	0
dense_3 (Dense)	(None, 7)	3591

```
1  from keras.models import Model, load_model
2  from keras.utils import plot_model
3
4  # read model
5  model_name = 'mcp-best-acc-0.68250.h5'
6  model = load_model(model_name)
7  model.summary()
```

⇨ 使用這個可以輕易觀察各層之間的關係

b. 利用 fit 的回傳參數可以將訓練過程的曲線畫出來:

取資料:

```
history = model.fit_generator(
    datagen.flow(train_feature,train_label,batch_size=128),
    steps_per_epoch=len(train_feature)/32,
    epochs=50,
    validation_data=(valid_feature,valid_label),
    callbacks=[mcp,es])
```

畫圖:

```
import matplotlib.pyplot as plt
import numpy as np

loss_cnn = np.load('./hw3_model/new_model/both_tra_loss.npy')
loss_dnn = np.load('./hw3_model/new_model/dnn_tra_loss.npy')
acc_cnn = np.load('./hw3_model/new_model/both_tra_acc.npy')
acc_dnn = np.load('./hw3_model/new_model/dnn_tra_acc.npy')



# plt.plot(loss_cnn,c='r',label='loss_cnn')
# plt.plot(loss_dnn,c='g',label='loss_dnn')
# plt.legend()
# plt.show()

plt.plot(acc_cnn,c='r',label='accuracy_cnn')
plt.plot(acc_dnn,c='g',label='accuracy_dnn')
plt.legend()
plt.show()
```