學號：B04501127 系級： 土木三 姓名：凌于凱

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?

模型架構：4* conv block → fully connected

conv block1: Conv2D(64, 5, 5) → Conv2D(64, 3, 3)
→MaxPooling(2, 2)

conv block2: Conv2D(128, 3, 3) → Conv2D(128, 3, 3)
→ Conv2D(128, 3, 3) → Conv2D(128, 3, 3)
→ MaxPooling(2, 2)

conv block3: Conv2D(256, 3, 3) → Conv2D(256, 3, 3)
→ Conv2D(256, 3, 3) → Conv2D(256, 3, 3)
→ MaxPooling(2, 2)

conv block1: Conv2D(512, 3, 3) → Conv2D(512, 3, 3)
→MaxPooling(2, 2)

fully connected: Dense(512) → Dense(512) → Dense(7)

dropout: 每一個 conv block 的 dropout rate 從 0.25 遞增至 0.5，fully connected 每層 dropout rate = 0.5

activation: LeakyRelu(alpha = 0.5)

詳細見 ./model.png

訓練參數：optimizer: adam(default)

validation split: 0.1 from training set

data augmentation: zoom_range=0.2, shear_range=0.2, rotation_range=30, width_shift_range=0.2, height_shift_range=0.2, horizontal_flip=True, vertical_flip=False

callbacks: 利用 ModelCheckpoint 儲存 val_acc 最高的 model（因為最一開始時用 EarlyStopping 成果並不是很好，常常 model 還沒 train 好就跳掉，就改成用 ModelCheckpoint）

batch_size: 128

epoch: 300（花費時間：392sec/epoch on GTX1060-6G）

準確率：public score: 0.69545, private score: 0.67539

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?

1)data normalization:

方法：將 x_train 和 x_test 合起來計算 48*48 個 mean, std 在將每一個 x 減去 mean 除以 std

影響：public score: 0.65645 → 0.66035

private score: 0.65951（跟沒有 normalization 一樣）

2)data augmentation:

方法：利用 ImageDataGenerator 增加經過翻轉，旋轉，尺寸變化，平移的圖片，並用 fit_generator 在每一個 epoch 都產生一定數量的 training data。

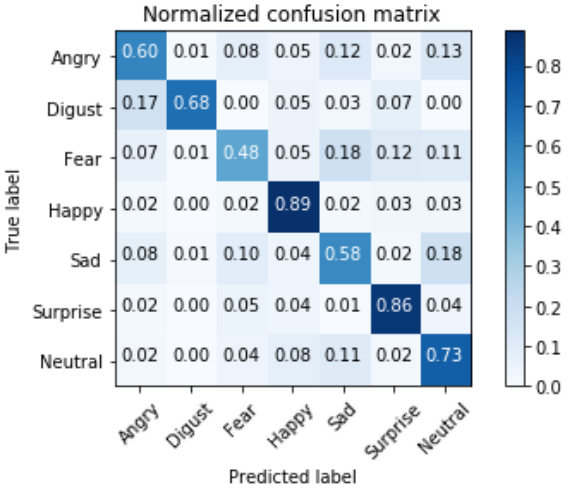
```
datagen = ImageDataGenerator(zoom_range=0.2,shear_range=0.2,rotation_range=30,
        width_shift_range=0.2, height_shift_range=0.2, horizontal_flip=True,
        vertical_flip=False)

history = model.fit_generator(datagen.flow(x_train,y_train,batch_size=128),
                              steps_per_epoch=int(np.ceil(5*x_train.shape[0] / 128.0)),
                              validation_data=(x_valid,y_valid),
                              epochs=200,verbose=2,
                              callbacks=callbacks_list)
```

影響：public score: 0.65645 → 0.69545

private score: 0.65951 → 0.67539

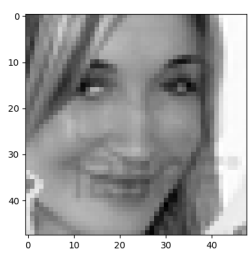
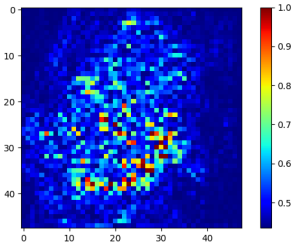
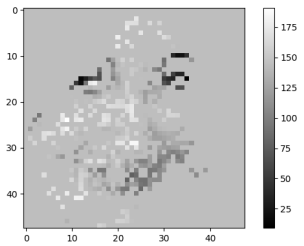
3. (1%) 觀察答錯的圖片中，哪些 class 彼此間容易用混? [繪出 confusion matrix 分析]

(參考: http://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html)



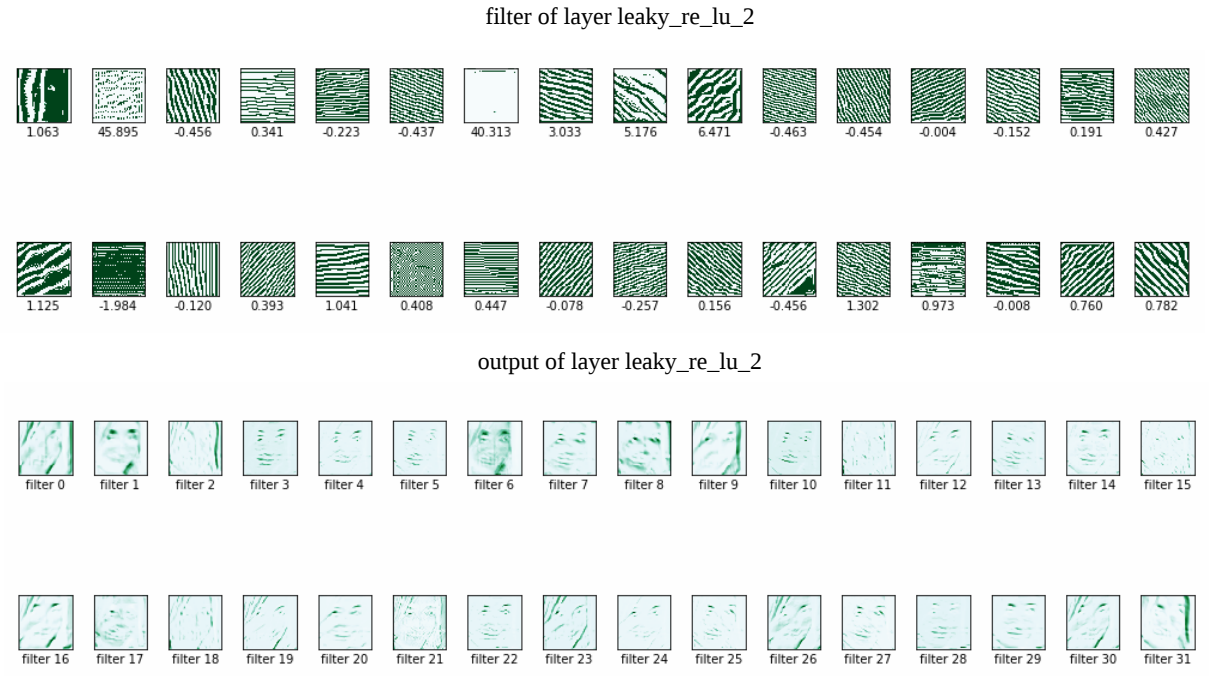
sad、fear 最容易搞混，可能是因為模型再做分類時主要 focus 在眼睛和嘴巴的部份，而負面情緒的表情通常在這兩部份很類似，以導致此結果。

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?

原圖	Saliency Map	Mask 掉 heat 小的部份
		

由上表可發現，經過 conv2D 和 MaxPooling 後，將對於分類沒有幫助的部份隱藏掉了，讓最後的 classification 主要 focus 在眼睛還有嘴巴的部份。

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。
(參考：https://blog.keras.io/how-convolutional-neural-networks-see-the-world.html)



從 input 跟 output 對照，我覺得 filter 主要觀察到的應該是眼睛和嘴巴，因為從 output 來看只有這兩個部份比較沒有被模糊掉，而 input filter 有些長的差不多但傾斜角度不同，我認為是 imageDataGenerator 的 rotate_range 所造成的，因為那些角度大都不大於 30 度。