學號：R06725028 系級： 資管碩一 姓名：黃于真

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？(Collaborators: r06725053)

使用 data normalization, data augmentation，並搭配以下參數和模型結構，在ep204時validation準確率達到最高峰0.70180，上傳後在kaggle上的public/private分數為0.69322/0.68626。

#global參數

epochs = 250

batch_size = 128

validation_split = 0.2

shuffle = True

```
datagen = ImageDataGenerator(rotation_range=30, width_shift_range=0.2, height_shift_range=0.2, \
    zoom_range=[0.8, 1.2], shear_range=0.2, horizontal_flip=True)
```

```
model = Sequential()
model.add(Conv2D(64, kernel_size=(5, 5), input_shape=(48,48,1),padding='same', kernel_initializer='
model.add(LeakyReLU(alpha=0.03))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2), padding='same'))
model.add(Dropout(0.3))



model.add(Conv2D(512, kernel_size=(3, 3),padding='same', kernel_initializer='glorot_normal'))
model.add(LeakyReLU(alpha=0.05))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2), padding='same'))
model.add(Dropout(0.5))

model.add(Conv2D(512, kernel_size=(3, 3),padding='same', kernel_initializer='glorot_normal'))
model.add(LeakyReLU(alpha=0.05))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2), padding='same'))
model.add(Dropout(0.5))



model.add(Conv2D(256, kernel_size=(3, 3),padding='same', kernel_initializer='glorot_normal'))
model.add(LeakyReLU(alpha=0.03))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2), padding='same'))
model.add(Dropout(0.5))

model.add(Conv2D(128, kernel_size=(3, 3),padding='same', kernel_initializer='glorot_normal'))
model.add(LeakyReLU(alpha=0.03))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2), padding='same'))
model.add(Dropout(0.5))
```

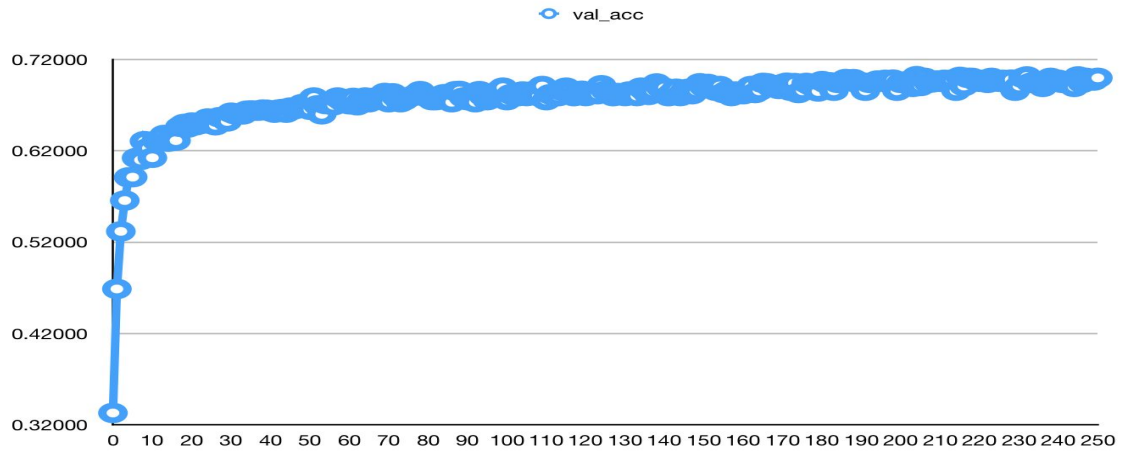
```
model.add(Flatten())

model.add(Dense(512, kernel_regularizer=l2(0), kernel_initializer='glorot_normal'))
model.add(Activation('relu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))

model.add(Dense(512, kernel_initializer='glorot_normal'))
model.add(Activation('relu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))

model.add(Dense(7, activation='softmax', kernel_initializer='glorot_normal'))
```

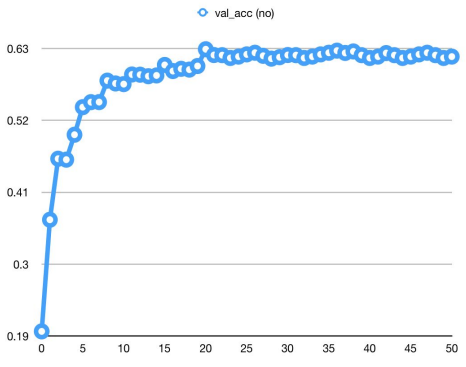
```
model.compile(optimizer='Adam',loss='categorical_crossentropy',metrics=['accuracy'])
```

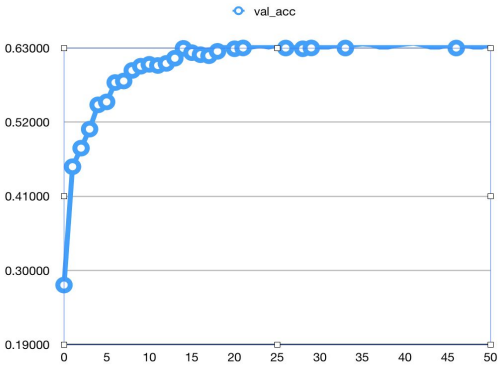


2. (1%) 請嘗試 data normalization, data augmentation,說明實行方法並且說明對準確率有什麼樣的影響？

[data normalization]

使用np.mean()、np.std()函數來計算特徵的mean和std，再用(x-mean)/std的方式來標準化特徵為mean=0、std=1，實際跑發現，使用相同參數與模型架構並訓練50 eppochs的情況下，有做標準化的模型在ep1的validation準確率就有0.2783，高峰為ep41的0.63689，而沒有做的在ep1的validation準確率只有0.1971，在ep20時就達到最高峰0.62887，可見有做標準化可以增加訓練效果，讓模型準確率增加。

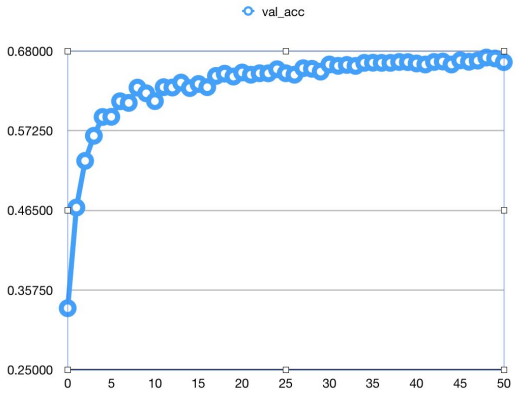


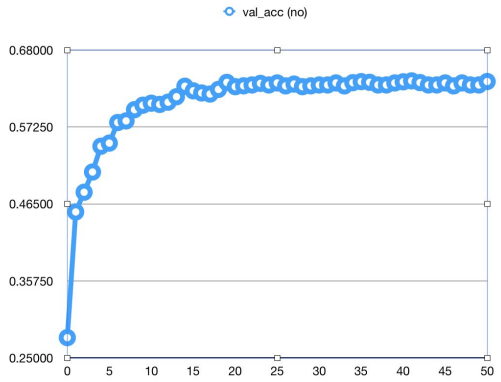


[data augmentation]

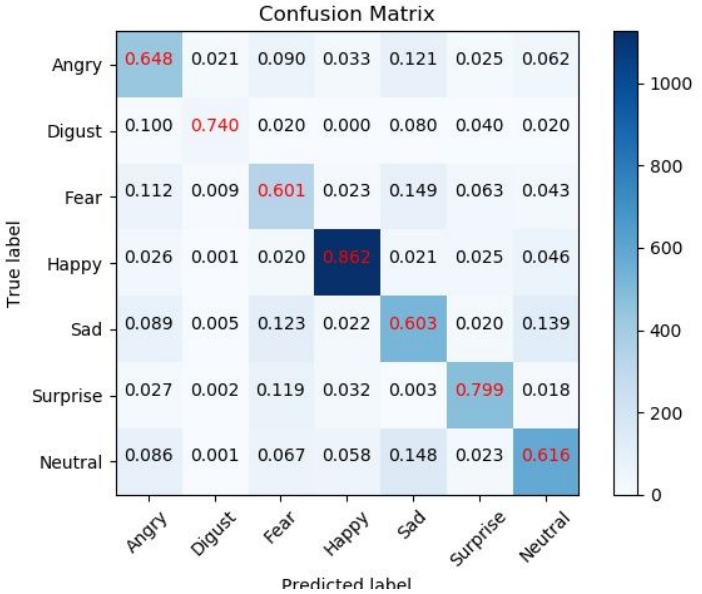
使用keras提供的ImageDataGenerator函數增加資料量，參數設定如第一小題的回答，實際跑發現，使用相同參數與模型架構並訓練50 eppochs的情況下，有沒有做data augmentation的準確率差非常大！有做的模型在ep1的validation準確率就有0.33300，高峰為ep48的0.67120，而沒有做的在ep1的validation準確率只有0.26942，高峰為ep41的0.63689，可見有做data augmentation來增加資料量，對模型的訓練很有幫助，可以避免overfitting，使得準確率提高許多，雖然一開始沒有做的準確率增長較快，但由於起跑點差太多，且如果再持續訓練下去，沒有做的也很難再提升準確率，會在0.63~0.64間

徘徊，但有做的就能持續增加，最後ep250的的validation準確率將近0.7。





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]

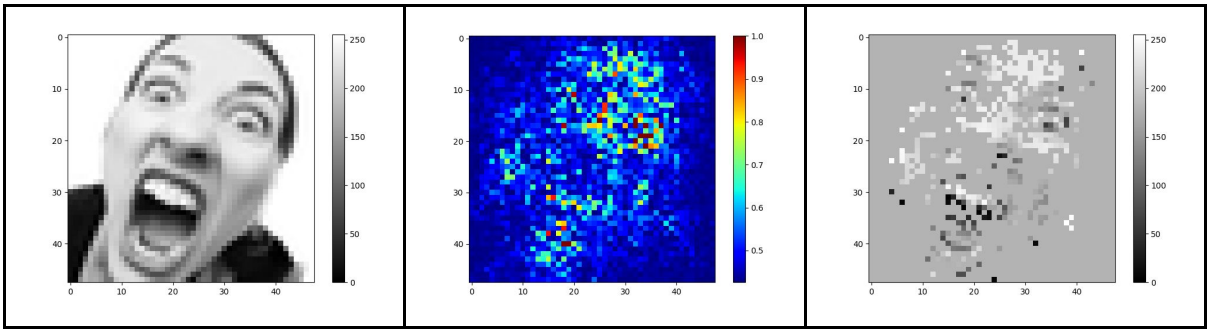


由上圖可知，模型最容易分錯的是fear和sad兩類，兩類最容易分錯的類也互為彼此。

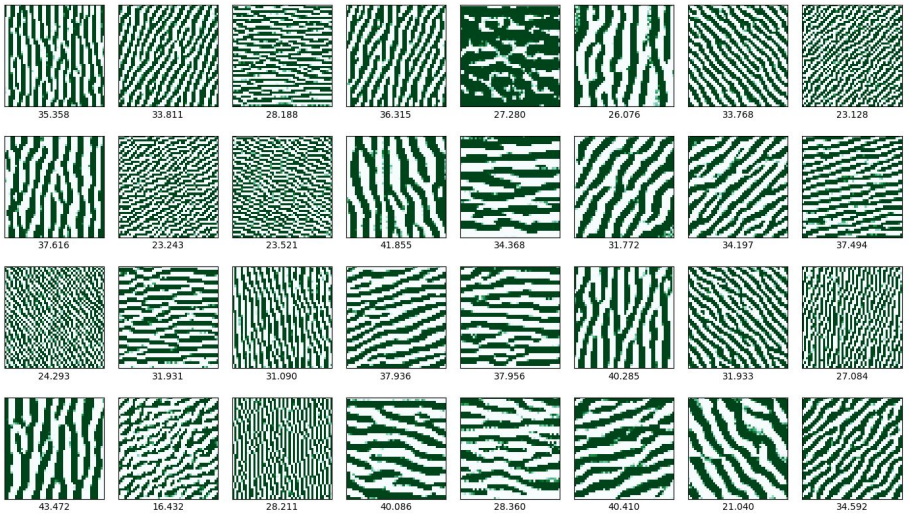
4. (1%) 從(1)(2)可以發現，使用 CNN 的確有些好處，試繪出其 saliency maps，觀察模型在做 classification 時，是 focus 在圖片的哪些部份？

這張圖被標記為angry，模型也預測正確，從Saliency Map可以看出，模型分類時觀察的是原圖中人誇張的五官，尤其是睜大的眼睛和張大的嘴巴，讓人感覺到此人的情緒是angry，所以在Mask掉heat小的圖中就發現隱約還可以看出眼睛和嘴巴，但其他部分則被模糊掉了。

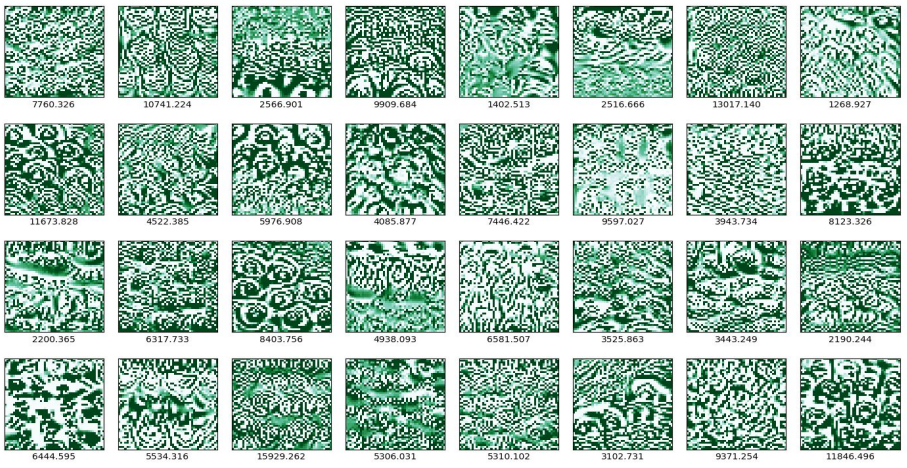
原圖	Saliency Map	Mask掉heat小的部份



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。



（圖a）



（圖b）

圖a為第一層leakyrelu的結果，圖b為第五層leakyrelu的結果，可以看到能active第一層的是比較簡單的圖案，包含不同方向、粗細不同的條紋，但能active第五層的圖案就比較複雜，出現許多漩渦的形狀，雖然還是抽象，但和第一層相比，似乎比較能和人臉做連結，這可能說明在第一層，模型判斷的是圖案中比較大範圍、大面積的線條趨勢，而到第五層，模型就會關注比較細節的圖案部分。