學號：B05902127 系級： 資工二 姓名：劉俊緯

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?

Layers	shape	parameters
conv2d_1 (Conv2D)	(None, 48, 48, 16)	channels=16
batch_normalization_1 (Batch	(None, 48, 48, 16)	momentum=0.5
leaky_re_lu_1 (LeakyReLU)	(None, 48, 48, 16)	(keras default=0.3)
conv2d_2 (Conv2D)	(None, 48, 48, 32)	channels=32
gaussian_noise_1 (GaussianNo	(None, 48, 48, 32)	gaussian noise=0.1
batch_normalization_2 (Batch	(None, 48, 48, 32)	momentum=0.5
leaky_re_lu_2 (LeakyReLU)	(None, 48, 48, 32)	(keras default=0.3)
conv2d_3 (Conv2D)	(None, 48, 48, 64)	channels=64
batch_normalization_3 (Batch	(None, 48, 48, 64)	momentum=0.5
leaky_re_lu_3 (LeakyReLU)	(None, 48, 48, 64)	(keras default=0.3)
max_pooling2d_1 (MaxPooling2	(None, 24, 24, 64)	
dropout_1 (Dropout)	(None, 24, 24, 64)	droprate=0.1
conv2d_4 (Conv2D)	(None, 24, 24, 128)	channels=128
batch_normalization_4 (Batch	(None, 24, 24, 128)	momentum=0.5
leaky_re_lu_4 (LeakyReLU)	(None, 24, 24, 128)	(keras default=0.3)
max_pooling2d_2 (MaxPooling2	(None, 12, 12, 128)	
dropout_2 (Dropout)	(None, 12, 12, 128)	droprate=0.2
conv2d_5 (Conv2D)	(None, 12, 12, 256)	channels=256
batch_normalization_5 (Batch	(None, 12, 12, 256)	momentum=0.5
leaky_re_lu_5 (LeakyReLU)	(None, 12, 12, 256)	(keras default=0.3)
max_pooling2d_3 (MaxPooling2	(None, 6, 6, 256)	
dropout_3 (Dropout)	(None, 6, 6, 256)	droprate=0.2
conv2d_6 (Conv2D)	(None, 6, 6, 512)	channels=512
batch_normalization_6 (Batch	(None, 6, 6, 512)	momentum=0.5
leaky_re_lu_6 (LeakyReLU)	(None, 6, 6, 512)	(keras default=0.3)
max_pooling2d_4 (MaxPooling2	(None, 3, 3, 512)	
dropout_4 (Dropout)	(None, 3, 3, 512)	droprate=0.2
flatten_1 (Flatten)	(None, 4608)	
dense_1 (Dense)	(None, 512)	
batch_normalization_7 (Batch	(None, 512)	momentum=0.5
leaky_re_lu_7 (LeakyReLU)	(None, 512)	(keras default=0.3)
dropout_5 (Dropout)	(None, 512)	droprate=0.5
dense_2 (Dense)	(None, 256)	
leaky_re_lu_8 (LeakyReLU)	(None, 256)	(keras default=0.3)
dense_3 (Dense)	(None, 7)	
activation_1 (Activation)	(None, 7)	softmax

Public Score	Private Score
0.70688	0.70047

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?

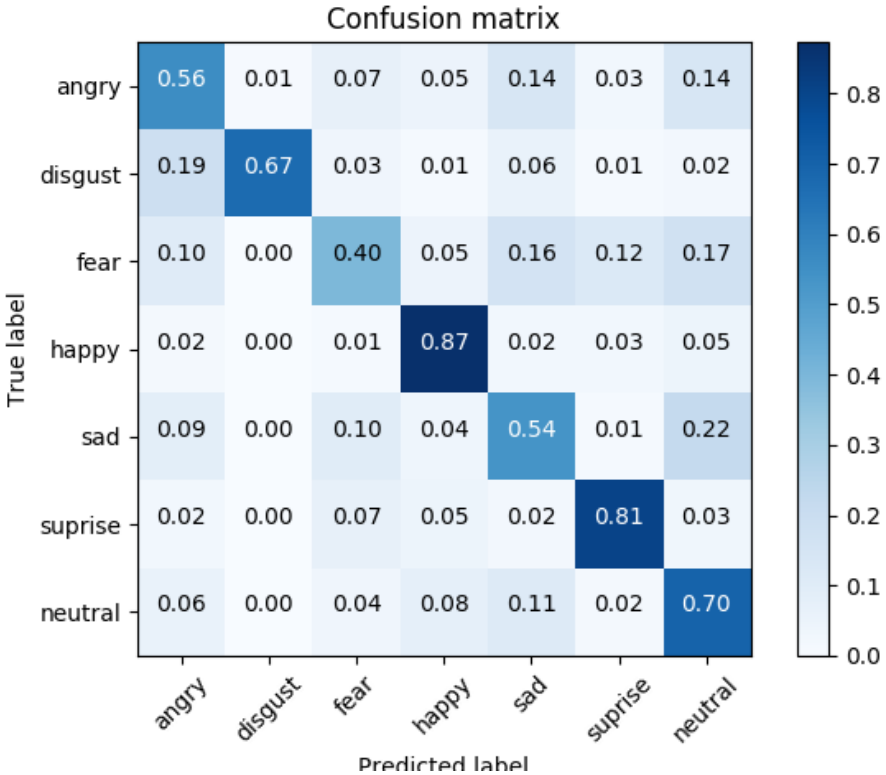
   Data augmentation 非常重要，因為可以大幅增加 dataset，使 overfitting 的情況減緩。至於 data normalization 其實沒有那麼重要，因為每個維度(48*48)都介於[0,255]，而且都是顏色，所以並沒有如其他 feature scaling 那麼重要。
   因此 normalization 準確率沒什麼提昇，而 augmentation 效果顯著。

	Without data normalization	With data normalization
Without data augmentation	0.58066 / 0.59152	0.59877 / 0.60769
With data augmentation	0.69434 / 0.70409	0.68737 / 0.69295

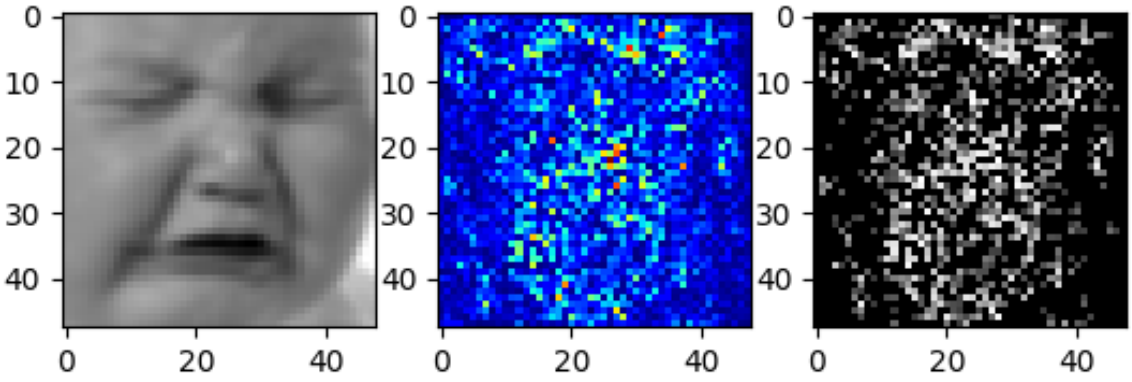
3. (1%) 觀察答錯的圖片中，哪些 class 彼此間容易用混?

我們看到 neutal-sad-fear 這三個類別的分數偏低，使得整體 accuracy 下降很多。這個非常符合我們直觀的感受。因為 sad / neutral 即使是以人來看，有些情況也難以分辨。至於 fear / sad 容易混淆也是一樣的道理。我們本來就難以用一張圖片去分別情緒。

至於 happy 容易分辨出來，可能是因為在 happy 的圖片中常有很多"露齒"，以及旁邊的嘴紋使得機器容易分辨。



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?



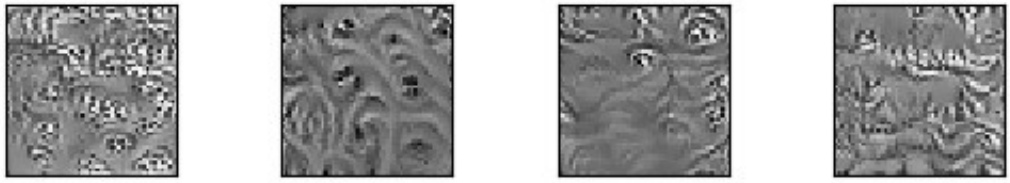
我們大致能夠看得出來，在這張 input 內，在眼睛上方，嘴巴附近的紋路還有鼻子附近有稍微的關注，我認為這個 model 比較關注在臉的紋路的地方，並用這些紋路推測出來是 sad 的依據，例如鼻子附近的紋路狀態能夠推測表情勢 sad/disgust。

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。

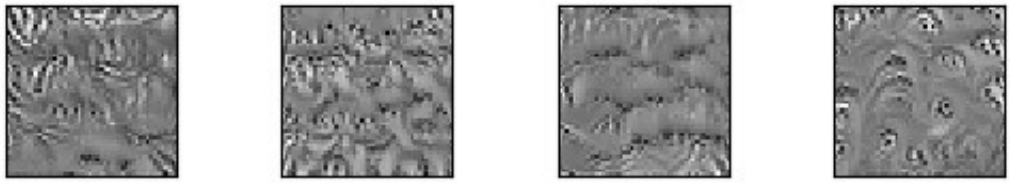
左：原本的圖。下 1：通過 Conv2d 第五層的 Filiter。

下 2：圖片經過 Conv2d_5 的 output。

經過了多層的轉換，
（我選擇第五層 conv 作為 CNN 視覺化的依據）

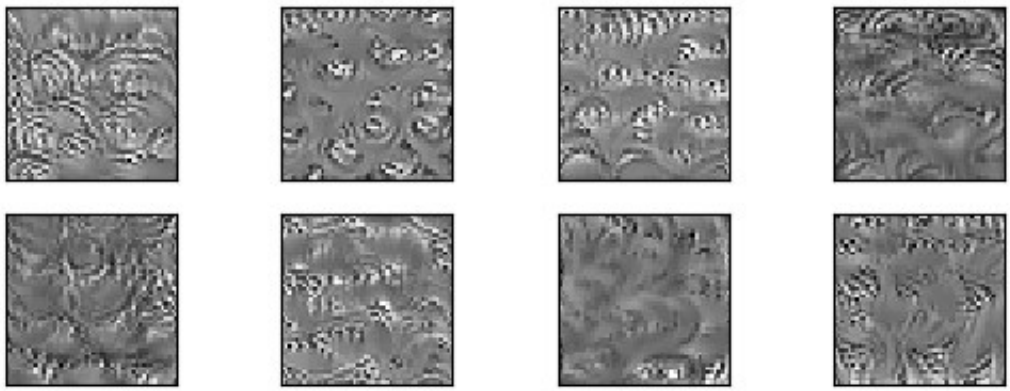


我們可以由右邊的圖得知這個 model 除了關住在線條之外，多關注了”洞”的特徵。



我們以人腦去思考，我認為最後從臉的紋路抓到洞的特徵不無道理，因為鼻孔，嘴巴，眼睛，都算是用洞就可以濾出來的特徵，

因此用 gradient ascent 做出來的圖呈現這樣的特徵其實算是正常的。



而我們用一張 input 來驗證上面的事情，我所選擇的 input 是第二面右下角的那一張。經過第五層的 conv 出來後的結果，我們觀察到眼睛，嘴巴，鼻子都有被 filter 出來，除了左上角的手可能有稍微的誤判，整體來說還是算有抓到東西的。
而我們觀察最後一張 output 又可以得知，在地五層的時候，還是有些 filter 是以”紋路”為關注焦點的。因而突顯出嘴巴附近的紋路。