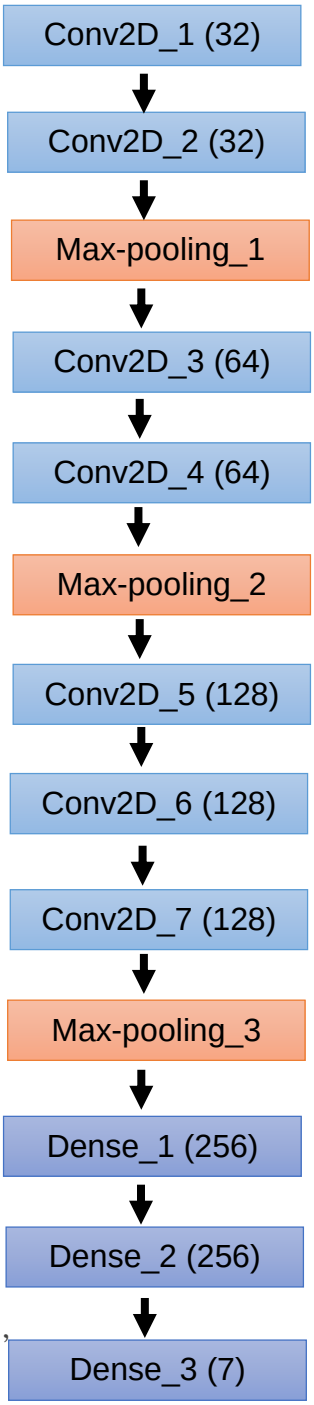
學號：R06921011 系級： 電機碩一 姓名：劉逸霖

1. (1%) 請說明你實作的 CNN model，其模型架構、訓練過程和準確率為何？
   答： Public : 0.68793 , Private : 0.68765 , Average : 0.68779

Layer (type)	Output Shape	Param #
zero_padding2d_1 (ZeroPaddin	(None, 50, 50, 1)	0
conv2d_1 (Conv2D)	(None, 48, 48, 32)	320
zero_padding2d_2 (ZeroPaddin	(None, 50, 50, 32)	0
conv2d_2 (Conv2D)	(None, 48, 48, 32)	9248
max_pooling2d_1 (MaxPooling2	(None, 24, 24, 32)	0
zero_padding2d_3 (ZeroPaddin	(None, 26, 26, 32)	0
conv2d_3 (Conv2D)	(None, 24, 24, 64)	18496
zero_padding2d_4 (ZeroPaddin	(None, 26, 26, 64)	0
conv2d_4 (Conv2D)	(None, 24, 24, 64)	36928
max_pooling2d_2 (MaxPooling2	(None, 12, 12, 64)	0
zero_padding2d_5 (ZeroPaddin	(None, 14, 14, 64)	0
conv2d_5 (Conv2D)	(None, 12, 12, 128)	73856
zero_padding2d_6 (ZeroPaddin	(None, 14, 14, 128)	0
conv2d_6 (Conv2D)	(None, 12, 12, 128)	147584
zero_padding2d_7 (ZeroPaddin	(None, 14, 14, 128)	0
conv2d_7 (Conv2D)	(None, 12, 12, 128)	147584
max_pooling2d_3 (MaxPooling2	(None, 6, 6, 128)	0
flatten_1 (Flatten)	(None, 4608)	0
dense_1 (Dense)	(None, 256)	1179904
dropout_1 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 256)	65792
dropout_2 (Dropout)	(None, 256)	0
dense_3 (Dense)	(None, 7)	1799

Total params: 1,681,511
Trainable params: 1,681,511
Non-trainable params: 0



除了這個架構之外，我使用 data augmentation 來 train，讓這個 model 不要那麼容易 overfit，因此我使用三個 datageneration，分別是旋轉 10 度/20 度/30 度，輪流 train，最後再使用原始 data(沒有旋轉)做最後的 fine tune，只跑 3 個 epochs 左右，就能達到 Kaggle 上 0.68 的準確率。

2. (1%) 承上題，請用與上述 CNN 接近的參數量，實做簡單的 DNN model。其模型架構、訓練過程和準確率為何？試與上題結果做比較，並說明你觀察到了什麼？

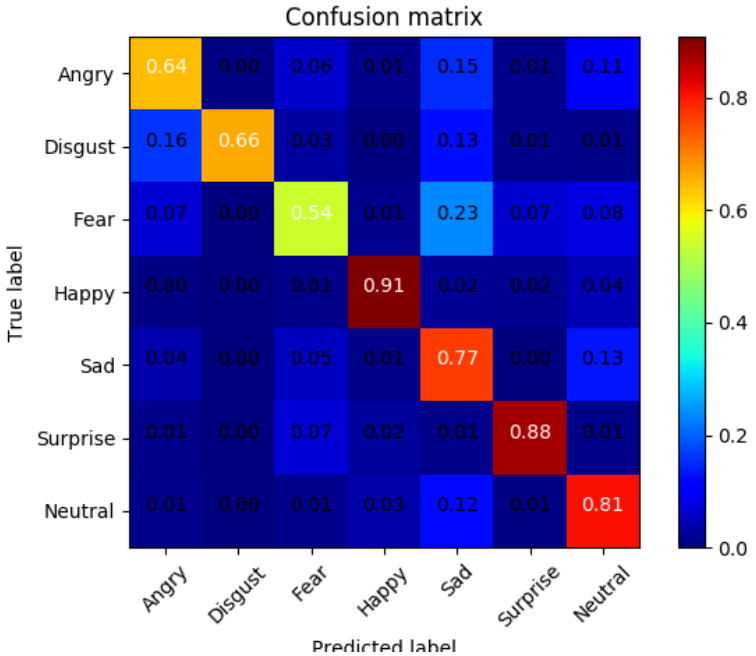
答：Public : 0.44441 , Private : 0.44106 , Average : 0.44273

Layer (type)	Output Shape	Param #
flatten_1 (Flatten)	(None, 2304)	0
dense_1 (Dense)	(None, 336)	774480
dense_2 (Dense)	(None, 336)	113232
dense_3 (Dense)	(None, 336)	113232
dense_4 (Dense)	(None, 336)	113232
dense_5 (Dense)	(None, 336)	113232
dense_6 (Dense)	(None, 336)	113232
dense_7 (Dense)	(None, 336)	113232
dense_8 (Dense)	(None, 336)	113232
dense_9 (Dense)	(None, 336)	113232
dense_10 (Dense)	(None, 7)	2359

Total params: 1,682,695
Trainable params: 1,682,695
Non-trainable params: 0

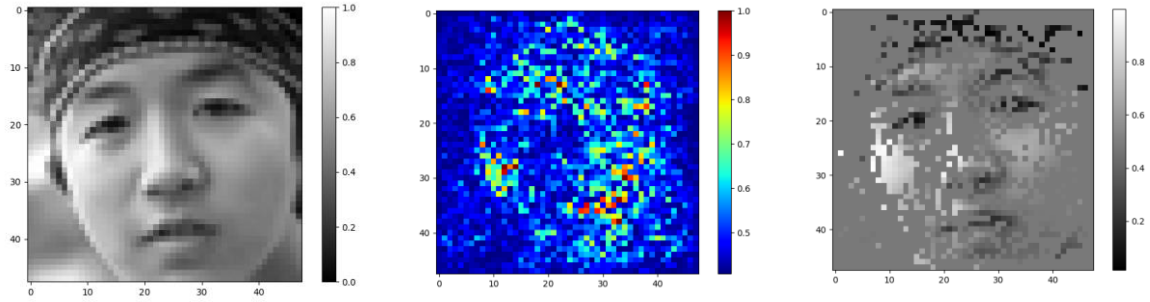
為了讓參數相同，層數也一樣(10 層)，算出每一層的 neuron 為 336 個 (除了 output layer 之外)。
從 training 過程可以發現，無論怎麼調參數或改變層數，準確率都在 0.40 附近，怎麼樣都上不去，試過最好的結果為 0.44，顯然 CNN 在影像識別上的效果比單純的 DNN 好很多。

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]



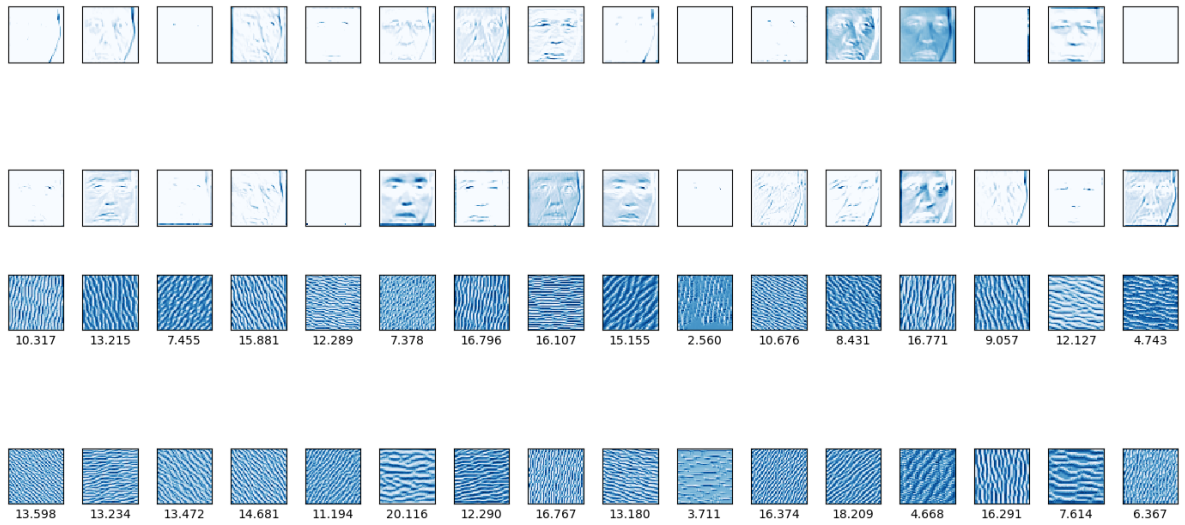
答：
因為我使用 data augmentation 來 train model 時，沒有切 validation，所以整個 confusion matrix 看起來的 accuracy 會偏高，不過依然可以觀察出這個 model 很容易把圖片誤判成 sad，尤其是 fear 的圖片有 23%的機率會判斷成 sad。而正確的圖片中，以 happy 的準確度最高，就我個人而言，我也是覺得開心的圖片應該會是最好判讀的。

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：



可以看出 model 在判斷表情時，著重的點是在五官的部分，眼睛鼻子及嘴巴，這跟我們人類的認知很接近。

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。
   答：



因為越後面的 CNN filter 提取出來的 feature 越難理解，看起來很不像人臉了，所以我取第二層來觀察。可以看出有些 filter 是用來找水平線(ex:第 5 個)，所以他找出了跟眼睛有關的特徵；有些 filter 是用來找鉛直線(ex:第 1 個)，所以他找出了臉的輪廓。