學號：B03902013 系級： 資工四　姓名：吳克駿

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？

Collaborators:張庭維、楊力權

答：模型架構

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 44, 44, 64)	1664
activation_1 (Activation)	(None, 44, 44, 64)	0
batch_normalization_1 (Batch	(None, 44, 44, 64)	256
zero_padding2d_1 (ZeroPaddin	(None, 48, 48, 64)	0
max_pooling2d_1 (MaxPooling2	(None, 22, 22, 64)	0
zero_padding2d_2 (ZeroPaddin	(None, 24, 24, 64)	0
dropout_1 (Dropout)	(None, 24, 24, 64)	0
conv2d_2 (Conv2D)	(None, 22, 22, 64)	36928
activation_2 (Activation)	(None, 22, 22, 64)	0
batch_normalization_2 (Batch	(None, 22, 22, 64)	256
zero_padding2d_3 (ZeroPaddin	(None, 24, 24, 64)	0
dropout_2 (Dropout)	(None, 24, 24, 64)	0
conv2d_3 (Conv2D)	(None, 22, 22, 128)	73856
activation_3 (Activation)	(None, 22, 22, 128)	0
batch_normalization_3 (Batch	(None, 22, 22, 128)	512
average_pooling2d_1 (Average	(None, 10, 10, 128)	0
zero_padding2d_4 (ZeroPaddin	(None, 12, 12, 128)	0
dropout_3 (Dropout)	(None, 12, 12, 128)	0
conv2d_4 (Conv2D)	(None, 10, 10, 128)	147584
activation_4 (Activation)	(None, 10, 10, 128)	0
batch_normalization_4 (Batch	(None, 10, 10, 128)	512
zero_padding2d_5 (ZeroPaddin	(None, 12, 12, 128)	0
dropout_4 (Dropout)	(None, 12, 12, 128)	0
conv2d_5 (Conv2D)	(None, 10, 10, 128)	147584
activation_5 (Activation)	(None, 10, 10, 128)	0
batch_normalization_5 (Batch	(None, 10, 10, 128)	512
zero_padding2d_5 (ZeroPaddin	(None, 12, 12, 128)	0
dropout_4 (Dropout)	(None, 12, 12, 128)	0
conv2d_5 (Conv2D)	(None, 10, 10, 128)	147584
activation_5 (Activation)	(None, 10, 10, 128)	0
batch_normalization_5 (Batch	(None, 10, 10, 128)	512
average_pooling2d_2 (Average	(None, 4, 4, 128)	0
zero_padding2d_6 (ZeroPaddin	(None, 6, 6, 128)	0
dropout_5 (Dropout)	(None, 6, 6, 128)	0
flatten_1 (Flatten)	(None, 4608)	0
dense_1 (Dense)	(None, 256)	1179904
activation_6 (Activation)	(None, 256)	0
dropout_6 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 128)	32896
activation_7 (Activation)	(None, 128)	0
dropout_7 (Dropout)	(None, 128)	0
dense_3 (Dense)	(None, 64)	8256
activation_8 (Activation)	(None, 64)	0
dropout_8 (Dropout)	(None, 64)	0
dense_4 (Dense)	(None, 7)	455
activation_9 (Activation)	(None, 7)	0

Total params: 1,631,175
Trainable params: 1,630,151
Non-trainable params: 1,024

訓練過程：

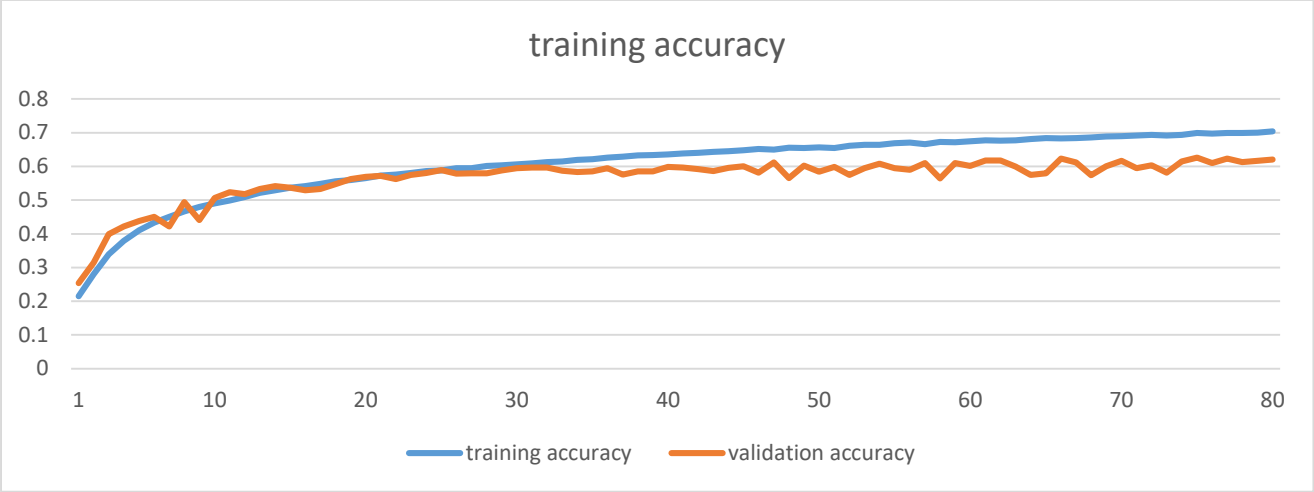
使用 categorical_crossentropy 來計算損失函數，優化器則使用 adadelta。

在 training_model 的部分，利用 np.rot90 來將圖片旋轉 90 度增加 data 數量。

取 50000 個 data train，最後面 7418 個 data 做 validation。

Epoch=80，batch=128，但會在中間儲存 validation accuracy 最高的 data。

訓練結果：



Kaggle 結果：public:60.378% private:61.186%

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？

Collaborators:張庭維、楊力權

答：

模型架構：

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 640)               1475200
_________________________________________________________________
activation_1 (Activation)    (None, 640)               0
_________________________________________________________________
dropout_1 (Dropout)          (None, 640)               0
_________________________________________________________________
dense_2 (Dense)              (None, 320)               205120
_________________________________________________________________
activation_2 (Activation)    (None, 320)               0
_________________________________________________________________
dropout_2 (Dropout)          (None, 320)               0
_________________________________________________________________
dense_3 (Dense)              (None, 160)               51360
_________________________________________________________________
activation_3 (Activation)    (None, 160)               0
_________________________________________________________________
dropout_3 (Dropout)          (None, 160)               0
_________________________________________________________________
dense_4 (Dense)              (None, 7)                 1127
_________________________________________________________________
activation_4 (Activation)    (None, 7)                 0
=================================================================
Total params: 1,732,807
Trainable params: 1,732,807
Non-trainable params: 0
_________________________________________________________________
```

過三層 DNN，總參數量比先前做的 CNN 稍多

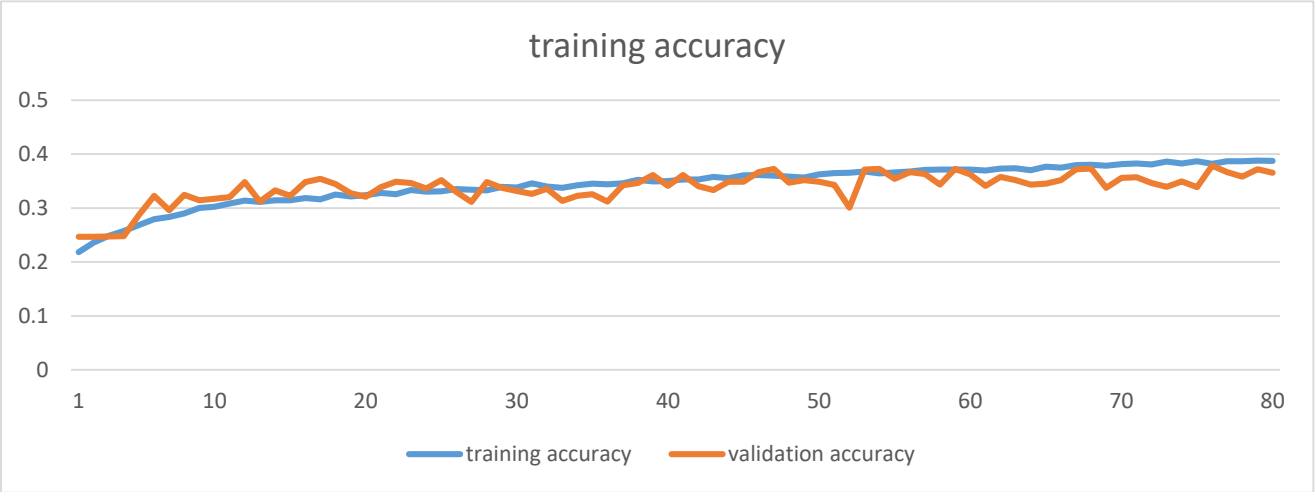
訓練過程：

直接將原本 2304 維的 data 餵入，epoch=80，batch=128。

一樣使用 categorical_crossentropy 來計算損失函數，優化器使用 adadelta。

沒有做圖片的翻轉，因此取前 25000 個 data 訓練，後面則做 validation

訓練結果：



Kaggle 結果：public:35.692% private:36.166%

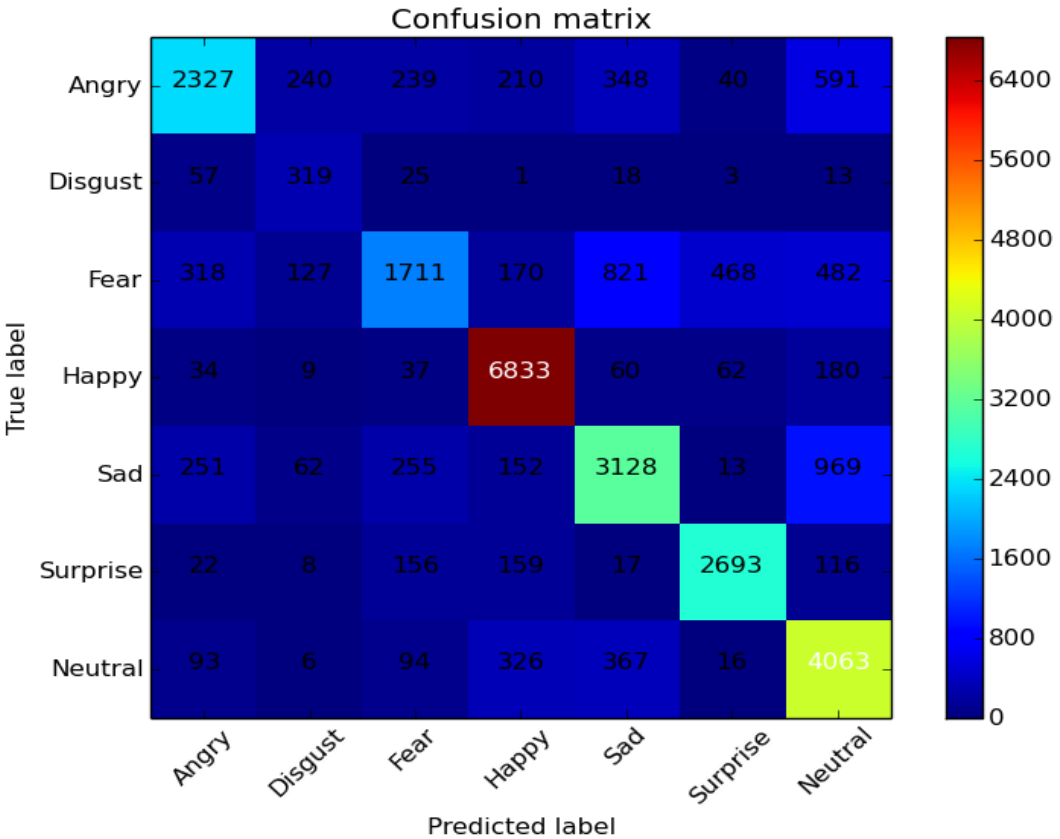
觀察：

由於 DNN 是鋪平的 data，因此圖片中上下的關係直接忽略，所以雖然使用的參數稍多一點，但做出的結果卻輸 CNN 不少。可能是因為 CNN 有 convolution 以及 pooling 的動作，更能抓出圖片的特徵，而在做 pooling 的同時，也有效的降低維度，因此也具提升效率的效果。

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]

Collaborators:None

答：

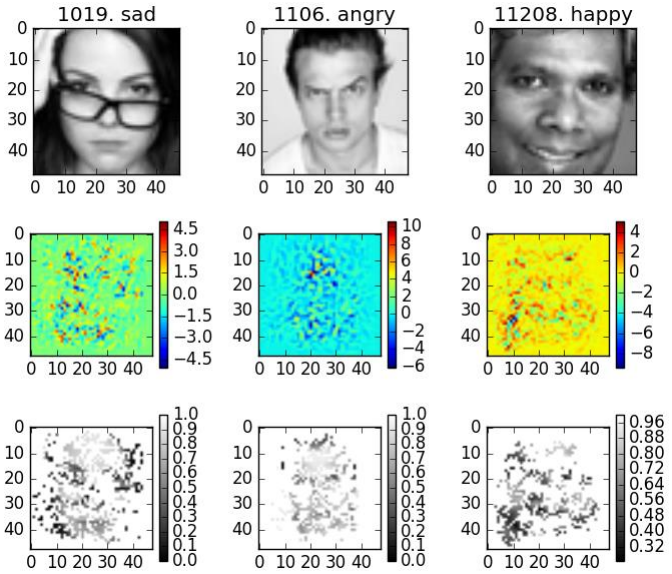


由圖可發現，由於 happy 的 model 最多，因此判斷起來的準確率也最高；而比例上較多判錯的則是 fear 跟 sad 之間，或者是把 label 錯判成 neutral，由於 neutral 很難定義，因此滿容易誤判。

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？

Collaborators: 張庭維、楊力權

答：

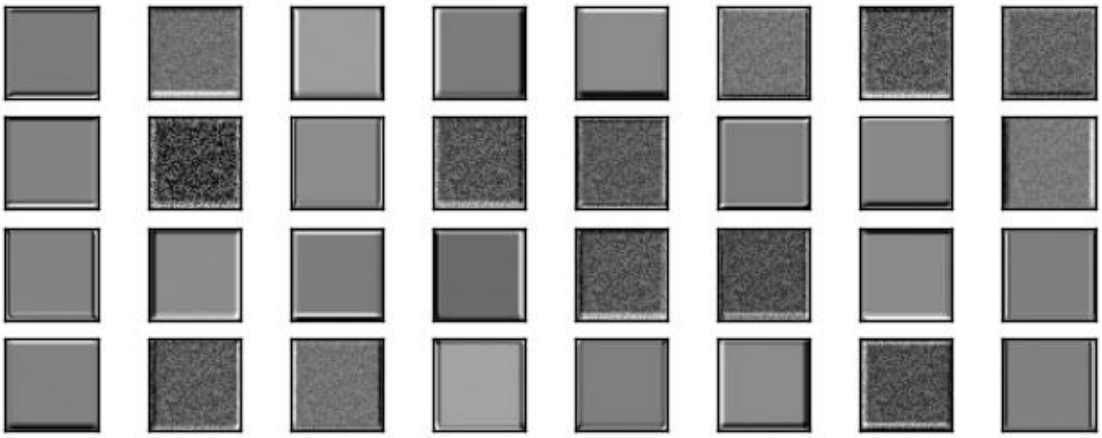


由圖可見，在做 classification 時，會比較 focus 在臉部的中間，尤其是五官為主，因為五官最能夠區別人的表情狀況。

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。

Collaborators: 張庭維、楊力權

答：



圖為對 CNN 第一層做 gradient accent，但幾乎所有的圖片都是雜訊，猜測是由於訓練才剛開始，且模型沒有訓練好，因此出現多雜訊。