學號：B05502145 系級： 電機三 姓名：林禹丞

1. (2%) 請說明你實作的 CNN model，其模型架構、訓練參數和準確率為何？並請用與上述 CNN 接近的參數量，實做簡單的 DNN model，同時也說明其模型架構、訓練參數和準確率為何？並說明你觀察到了什麼？
   (Collaborators: None)

   CNN 的模型架構：

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 46, 46, 64)	640
batch_normalization_1 (Batch	(None, 46, 46, 64)	256
max_pooling2d_1 (MaxPooling2	(None, 23, 23, 64)	0
dropout_1 (Dropout)	(None, 23, 23, 64)	0
conv2d_2 (Conv2D)	(None, 21, 21, 128)	73856
batch_normalization_2 (Batch	(None, 21, 21, 128)	512
max_pooling2d_2 (MaxPooling2	(None, 11, 11, 128)	0
dropout_2 (Dropout)	(None, 11, 11, 128)	0
conv2d_3 (Conv2D)	(None, 9, 9, 512)	590336
batch_normalization_3 (Batch	(None, 9, 9, 512)	2048
max_pooling2d_3 (MaxPooling2	(None, 5, 5, 512)	0
dropout_3 (Dropout)	(None, 5, 5, 512)	0
conv2d_4 (Conv2D)	(None, 3, 3, 512)	2359808
batch_normalization_4 (Batch	(None, 3, 3, 512)	2048
max_pooling2d_4 (MaxPooling2	(None, 2, 2, 512)	0
dropout_4 (Dropout)	(None, 2, 2, 512)	0
flatten_1 (Flatten)	(None, 2048)	0
dense_1 (Dense)	(None, 512)	1049088
batch_normalization_5 (Batch	(None, 512)	2048
dropout_5 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 512)	262656

```
batch_normalization_6 (Batch (None, 512)               2048
_________________________________________________________________
dropout_6 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_3 (Dense)              (None, 7)                 3591
=================================================================
Total params: 4,348,935
Trainable params: 4,344,455
Non-trainable params: 4,480
_________________________________________________________________
```

DNN 的模型架構：

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 512)               1180160
_________________________________________________________________
batch_normalization_1 (Batch (None, 512)               2048
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 512)               262656
_________________________________________________________________
batch_normalization_2 (Batch (None, 512)               2048
_________________________________________________________________
dropout_2 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_3 (Dense)              (None, 512)               262656
_________________________________________________________________
batch_normalization_3 (Batch (None, 512)               2048
_________________________________________________________________
dropout_3 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_4 (Dense)              (None, 1024)              525312
_________________________________________________________________
batch_normalization_4 (Batch (None, 1024)              4096
_________________________________________________________________
dropout_4 (Dropout)          (None, 1024)              0
_________________________________________________________________
dense_5 (Dense)              (None, 1024)              1049600
_________________________________________________________________
batch_normalization_5 (Batch (None, 1024)              4096
_________________________________________________________________
dropout_5 (Dropout)          (None, 1024)              0
_________________________________________________________________
dense_6 (Dense)              (None, 1024)              1049600
_________________________________________________________________
batch_normalization_6 (Batch (None, 1024)              4096
_________________________________________________________________
dropout_6 (Dropout)          (None, 1024)              0
_________________________________________________________________
dense_7 (Dense)              (None, 7)                 7175
=================================================================
Total params: 4,355,591
Trainable params: 4,346,375
```

```
Non-trainable params: 9,216
_________________________________________________________________
```

CNN 準確率：

Private Score	Public Score
0.69267	0.68821

DNN 準確率：

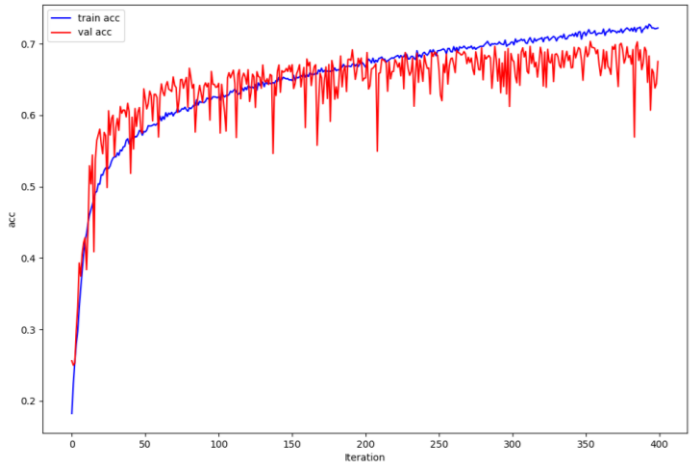
Private Score	Public Score
0.41264	0.43466

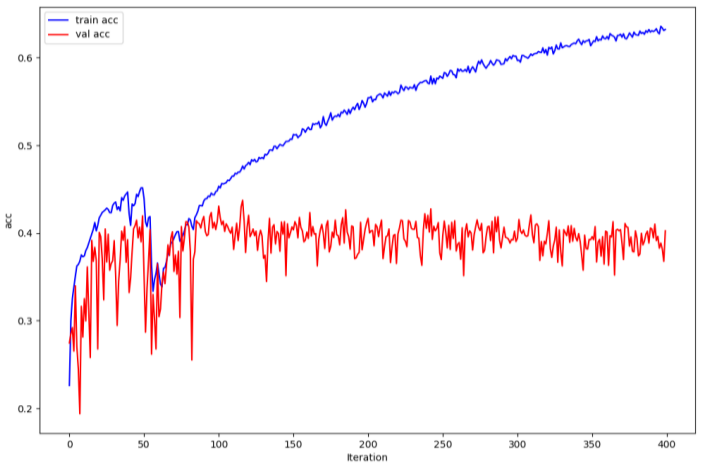
我們可以發現儘管 CNN 和 DNN 參數量差不多，DNN 在 training set 上的準確率很明顯比較低。

這個問題的主要原因是因為 CNN 可以抽取 feature，例如人的臉偏左邊和偏右邊一樣可以被 CNN 抽出 feature，而 DNN 是以每個 pixel 硬記的概念去 train，導致 DNN 沒辦法在影像辨識中有良好的結果。

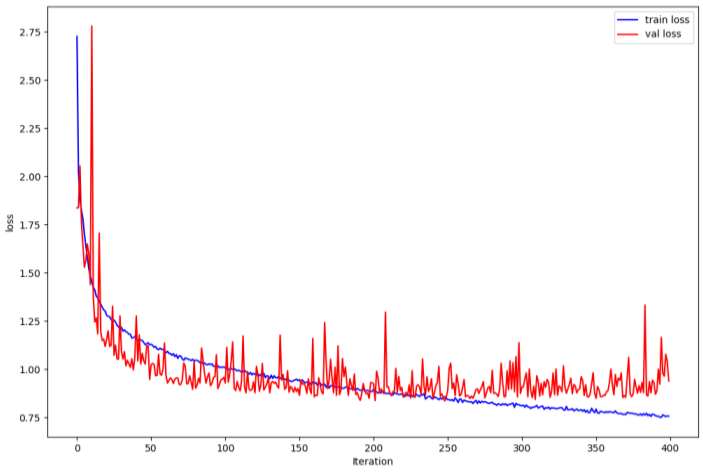
2. (1%) 承上題，請分別畫出這兩個 model 的訓練過程 (i.e., loss/accuracy v.s. epoch) (Collaborators: None)

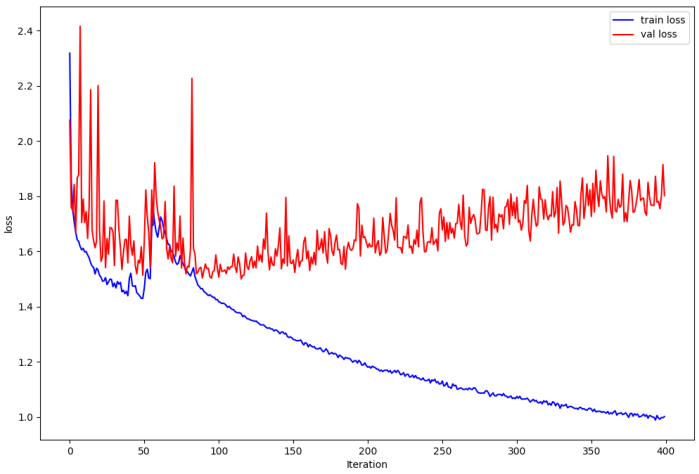
CNN & DNN 的 training accuracy v.s. epoch.：





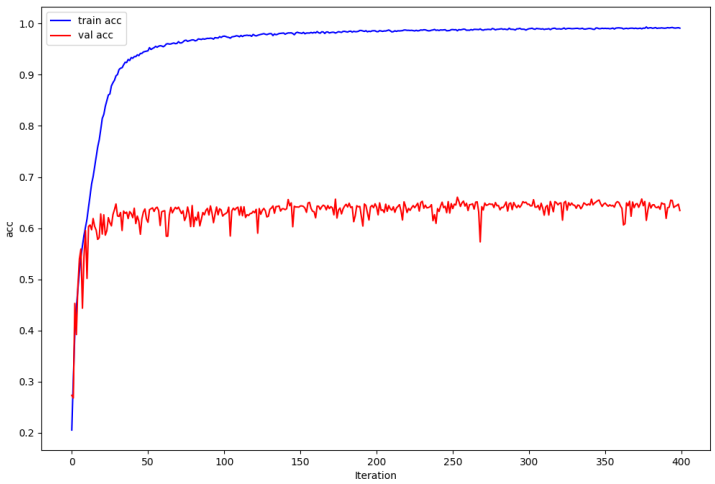
CNN & DNN 的 training loss v.s. epoch.：

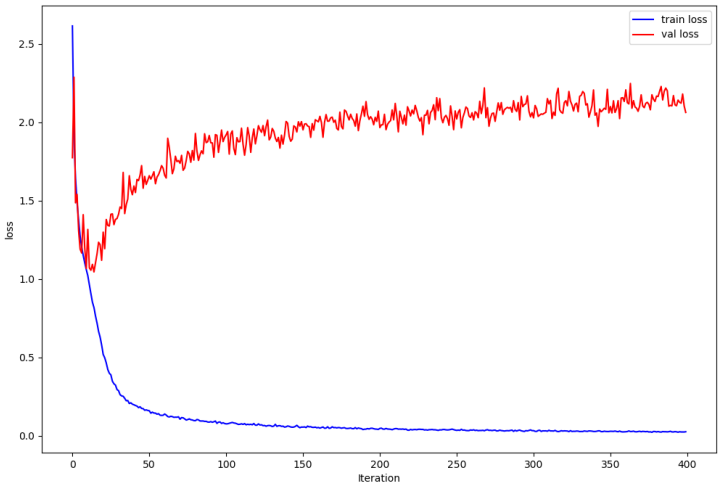




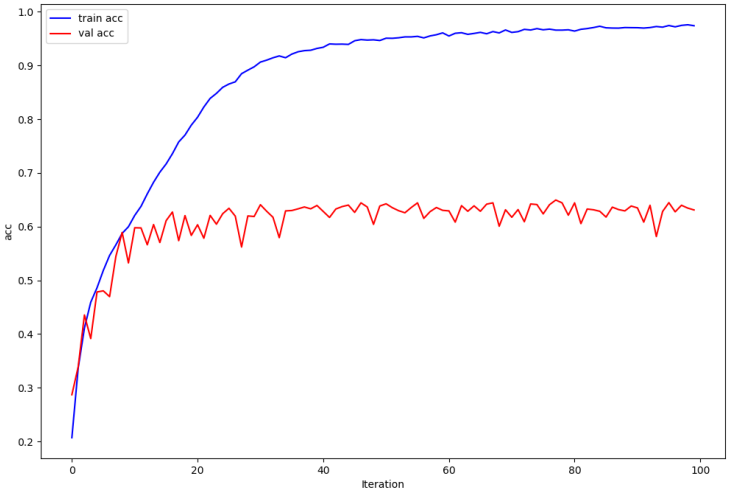
3. (1%) 請嘗試 data normalization, data augmentation,說明實作方法並且說明實行前後對準確率有什麼樣的影響？
(Collaborators: None)

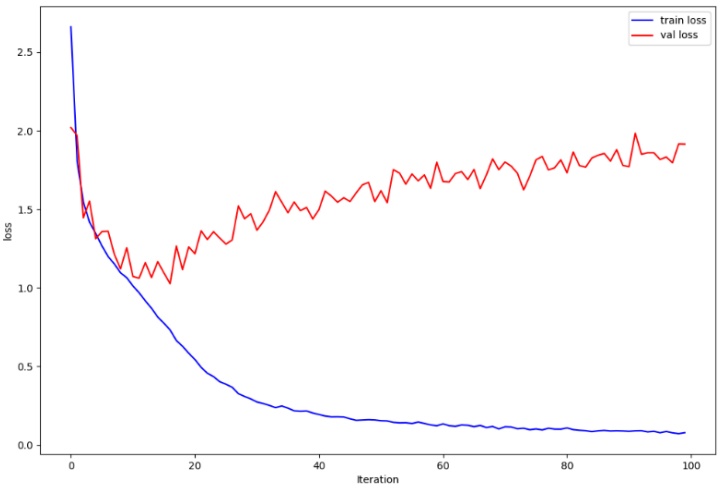
Model1(no normalize & no augmentation):



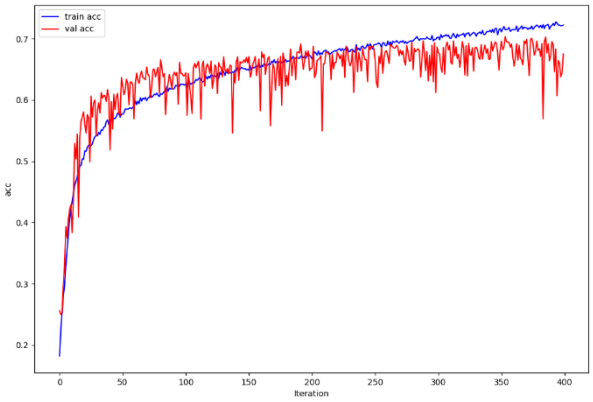


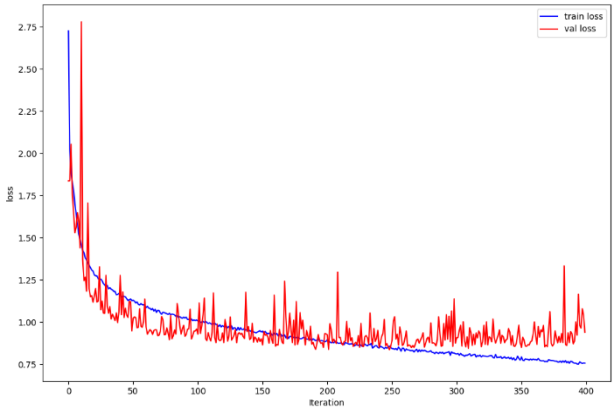
Model2 (normalize & no augmentation):





Model3: normalize & augmentation





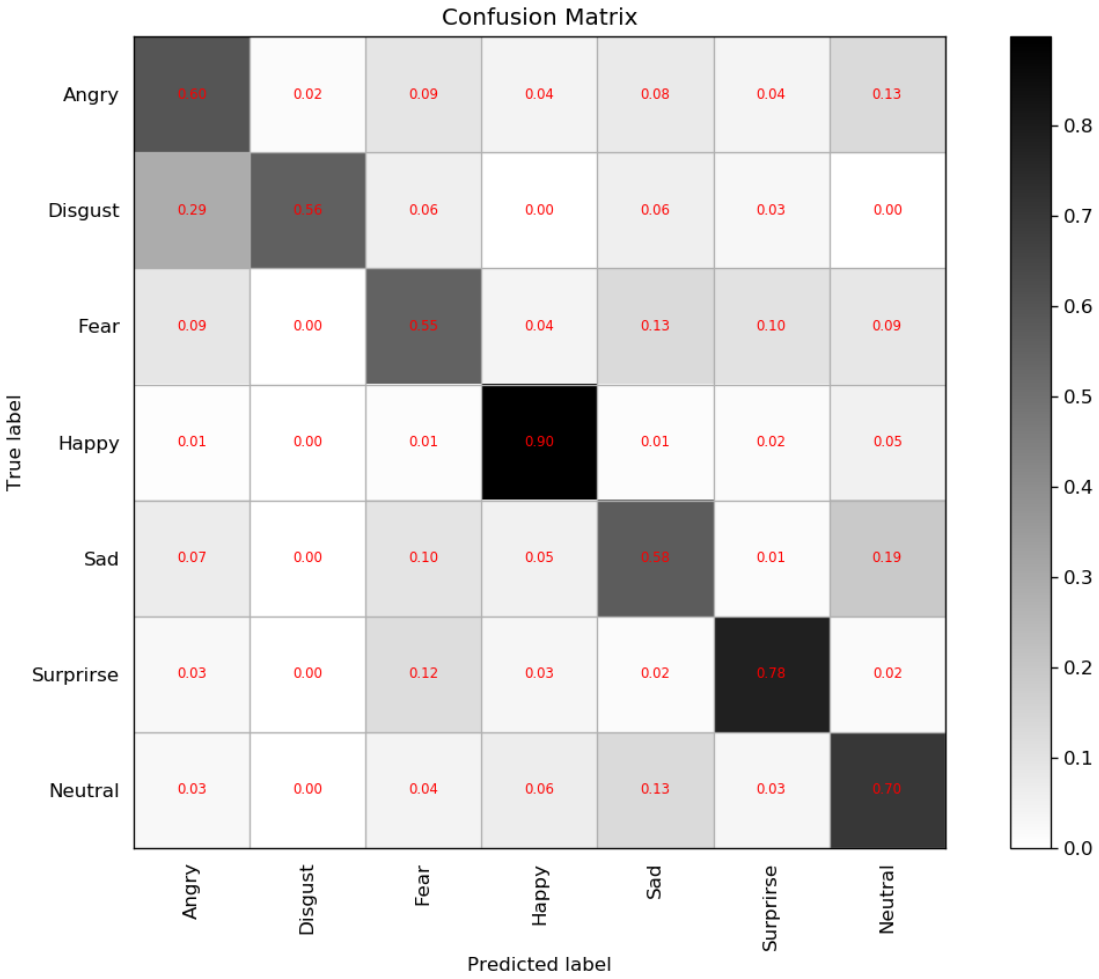
由上面三張圖我們可以發現，有無 normalize 對 training 和準確率影響不大。而 data augmentation 對準確率有大量的提升(epoch 到 300 之後可以發現和 Model1 準確率可以差到 5%)

關於 normalize 的實作方法，我們直接將 48*48*1 的 nparray 除以 255

關於 data augmentation 的實作方法，我們利用 keras 的 ImageDataGenerator，可以隨機對圖片進行旋轉、縮放、翻轉等。

4. (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 confusion matrix 可以看出：傷心(Sad)很容易被便認為中立(Neutral)和；而開心(Happy)則比較少被誤認為其他類別