Machine Learning HW3 Report

學號：B03901015 系級： 電機四 姓名：梅希聖

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: 自己、keras 的 VGG16/19)
   架構與準確率：
   參考 VGG16 的架構，每 2~3 層 convolution 層搭配一層 pooling，共 3 次，之後將 convolution 層之結果壓平後連接 2 層全連接層，再透過 softmax 輸出得到最後分類結果。其中有插入數層的 batch normalization 與 dropout 層調整模型。

Layer (type)	Output Shape	Param #
conv2d_129 (Conv2D)	(None, 48, 48, 32)	320
conv2d_130 (Conv2D)	(None, 48, 48, 32)	9248
batch_normalization_49 (Batc	(None, 48, 48, 32)	128
dropout_30 (Dropout)	(None, 48, 48, 32)	0
max_pooling2d_49 (MaxPooling	(None, 24, 24, 32)	0
conv2d_131 (Conv2D)	(None, 24, 24, 64)	18496
conv2d_132 (Conv2D)	(None, 24, 24, 64)	36928
conv2d_133 (Conv2D)	(None, 24, 24, 64)	36928
batch_normalization_50 (Batc	(None, 24, 24, 64)	256
dropout_31 (Dropout)	(None, 24, 24, 64)	0
max_pooling2d_50 (MaxPooling	(None, 12, 12, 64)	0
conv2d_134 (Conv2D)	(None, 12, 12, 128)	73856
conv2d_135 (Conv2D)	(None, 12, 12, 128)	147584
conv2d_136 (Conv2D)	(None, 12, 12, 128)	147584
batch_normalization_51 (Batc	(None, 12, 12, 128)	512
dropout_32 (Dropout)	(None, 12, 12, 128)	0
max_pooling2d_51 (MaxPooling	(None, 6, 6, 128)	0
flatten_17 (Flatten)	(None, 4608)	0
dense_41 (Dense)	(None, 64)	294976
p_re_lu_18 (PReLU)	(None, 64)	64
dropout_33 (Dropout)	(None, 64)	0
dense_42 (Dense)	(None, 64)	4160
p_re_lu_19 (PReLU)	(None, 64)	64
dropout_34 (Dropout)	(None, 64)	0
dense_43 (Dense)	(None, 7)	455

Total params: 771,559
Trainable params: 771,111
Non-trainable params: 448

然而，使用單一個 CNN model 訓練的準確度最高只有達到 0.673(public) / 0.661(private)，因此採用 ensemble 的方式，將 6~7 個類似架構的 CNN model 之預測結果進行平均，最後最高的分數到達 0.713(public) / 0.707(private)。

參數：
convolution layer 的 size 分別是 32, 64, 128，使用 relu 作為激化函數。
每個 CNN model 有些許不同，隨機選取 1/10 或 2/10 的 data 作為 validation，全連接層有部分 model 使用 prelu（比 relu 多了一個學習的參數），其他一樣使用 relu，
batch_size = 128，train epoch = 200 or 250。

- 2~5 題之實作皆採用自己的最好的單個 CNN model 進行測試（非 ensemble model）

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？
(Collaborators: 自己)
data normalization:
使用 sklearn 的 StandardScaler 套件實作 normalization，調整至 mean=0, std=1。

```
# image data stored in X
X = X.ravel() # reshape to 1D
scaler = StandardScaler()
scaler.fit(X)
X = scaler.transform(X)
X.reshape((7178,48*48))
```

	Private	Public	Average
With normalization	0.63750	0.64892	0.64321
Without normalization	0.64196	0.64892	0.64544

結果顯示，有無做 normalization 對我的 model 結果影響差異不大。

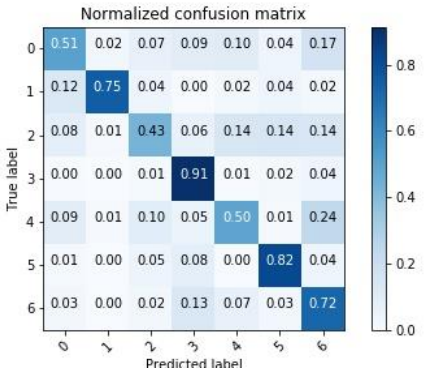
data augmentation:
使用 keras 的 ImageDataGenerator 套件實作 augmentation，套件會將圖片作旋轉、左右平移、水平翻轉、拉近拉遠等。

```
# Augmentation on image data
train_datagen = ImageDataGenerator(rotation_range=20,
                                   width_shift_range=0.1,
                                   height_shift_range=0.1,
                                   horizontal_flip=True,
                                   shear_range=0.2,
                                   zoom_range=0.25)
```

	Private	Public	Average
With augmentation	0.63750	0.64892	0.64321
Without augmentation	0.59431	0.59403	0.59417

結果顯示，使用 augmentation 會讓準確率提升許多，我的最佳 model 中也有使用 augmentation。

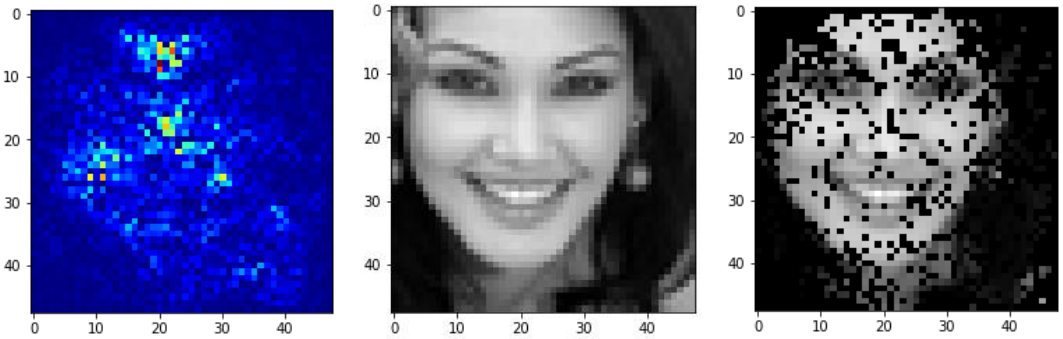
3. (1%) 觀察答錯的圖片中，哪些 class 彼此間容易用混？[繪出 confusion matrix 分析] (Collaborators: 自己)



從資料中取出 10%作為 validation，其餘進行 training。
結果顯示此模型在類別 3（高興）有最高的準確率(91%)，在類別 2（恐懼）的準確率最低(43%)，容易被誤認為其他類別，可能是我的 model 沒辦法擷取到恐懼圖片中之共同特徵。另外，類別 4（難過）的資料有 24%的機率被誤認成類別 6（中立），也是這個 model 的弱點。

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: 自己、keras-vis)

本題使用上題 validation set 中的其中一張圖片作測試，左為 saliency map，中為原始圖片，右為經過 filter 後的圖片。可發現梯度在人臉的部分較大，而在背景或是頭髮的部分較低，推測模型是正確地根據人臉的部分進行表情辨識。



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。
(Collaborators: https://blog.keras.io/how-convolutional-neural-networks-see-the-world.html)

model 之架構已在問題 1 呈述。
由左至右分別是 conv2d_130, conv2d_133, conv2d_136 之結果，3 個 layer 分別是 3 組 convolution filter 之最後一層。
上排為 input noise 後 filter 之結果，下排為 input image+noise 之結果。
觀察結果可發現，愈上層之 filter 所濾出的應是較大面積的特徵，愈下層的 layer，output 圖片愈細緻，應代表愈深層的 filter 可以看出更細節的東西。