ML2018 HW3 Report

學號：B05611033 系級： 生機二 姓名：杜杰翰

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:)

我使用了 7 個 model 進行 ensemble，每個都使用同樣的 data augmentation 與 normalization。

以下為第一個 model

```
dropout_rate = 0.2
model = Conv2D(60, (5, 5), input_shape= (48, 48, 1), activation='relu')(inputs)
model = Dropout(dropout_rate)(model)
model = MaxPooling2D((2, 2))(model)
model = Conv2D(120, (3, 3), activation='relu')(model)
model = Dropout(dropout_rate)(model)
model = MaxPooling2D((2, 2))(model)
model = Conv2D(240, (3, 3), activation='relu')(model)
model = MaxPooling2D((2, 2))(model)

model = Flatten()(model)
model = Dropout(dropout_rate)(model)
model = Dense(400, activation='relu')(model)
model = Dropout(dropout_rate)(model)
model = Dense(200, activation='relu')(model)
model = Dense(7, activation='softmax')(model)
model = Model(inputs=inputs, outputs=model)
```

第二個

```
dropout_rate = 0.2
model = Conv2D(60, (5, 5), input_shape= (48, 48, 1), activation='relu')(inputs)
model = Dropout(dropout_rate)(model)
model = MaxPooling2D((2, 2))(model)
model = Conv2D(120, (3, 3), activation='relu')(model)
model = Dropout(dropout_rate)(model)
model = MaxPooling2D((2, 2))(model)
model = Conv2D(240, (3, 3), activation='relu')(model)
model = MaxPooling2D((2, 2))(model)

model = Flatten()(model)
model = Dropout(dropout_rate)(model)
model = Dense(400, activation='relu')(model)
model = Dropout(dropout_rate)(model)
model = Dense(200, activation='relu')(model)
model = Dense(100, activation='relu')(model)
model = Dense(7, activation='softmax')(model)
model = Model(inputs=inputs, outputs=model)
```

第三個

```
dropout_rate = 0.2
model = Conv2D(60, (5, 5), input_shape= (48, 48, 1), activation='relu')(inputs)
model = Dropout(dropout_rate)(model)
model = MaxPooling2D((2, 2))(model)
model = Conv2D(120, (3, 3), activation='relu')(model)
model = Dropout(dropout_rate)(model)
model = MaxPooling2D((2, 2))(model)
model = Conv2D(240, (3, 3), activation='relu')(model)
model = MaxPooling2D((2, 2))(model)

model = Flatten()(model)
model = Dropout(dropout_rate)(model)
model = Dense(300, activation='relu')(model)
model = Dropout(dropout_rate)(model)
model = Dense(300, activation='relu')(model)
model = Dense(300, activation='relu')(model)
model = Dropout(dropout_rate)(model)
model = Dense(300, activation='relu')(model)
model = Dense(300, activation='relu')(model)
model = Dense(7, activation='softmax')(model)
model = Model(inputs=inputs, outputs=model)
```

第四個與第五個使用了 inception

第四個

```
dropout_rate = 0.2

model1 = Conv2D(60, (3, 3), activation='relu')(inputs)
model1 = Conv2D(60, (3, 3), activation='relu')(model1)
model1 = MaxPooling2D(2, 2)(model1)
model1 = Conv2D(120, (3, 3), activation='relu')(model1)
model1 = MaxPooling2D(2, 2)(model1)
model1 = Conv2D(240, (3, 3), activation='relu')(model1)
model1 = Flatten()(model1)

model2 = MaxPooling2D(2, 2)(inputs)
model2 = Conv2D(60, (3, 3), activation='relu')(model2)
model2 = MaxPooling2D(2, 2)(model2)
model2 = Conv2D(120, (3, 3), activation='relu')(model2)
model2 = MaxPooling2D(2, 2)(model2)
model2 = Flatten()(model2)

model = concatenate([model1, model2])

model = Dense(300, activation='relu')(model)
model = Dropout(dropout_rate)(model)
model = Dense(300, activation='relu')(model)
model = Dropout(dropout_rate)(model)
model = Dense(300, activation='relu')(model)
model = Dropout(dropout_rate)(model)
model = Dense(300, activation='relu')(model)
model = Dropout(dropout_rate)(model)
model = Dense(300, activation='relu')(model)
model = Dense(7, activation='softmax')(model)

model = Model(inputs=inputs, outputs=model)
```

第五個

```
dropout_rate = 0.25

model1 = Conv2D(60, (3, 3), activation='relu')(inputs)
model1 = Conv2D(60, (3, 3), activation='relu')(model1)
model1 = MaxPooling2D(2, 2)(model1)
model1 = Conv2D(120, (3, 3), activation='relu')(model1)
model1 = MaxPooling2D(2, 2)(model1)
model1 = Conv2D(240, (3, 3), activation='relu')(model1)
model1 = MaxPooling2D(2, 2)(model1)
model1 = Flatten()(model1)

model2 = MaxPooling2D(2, 2)(inputs)
model2 = Conv2D(80, (3, 3), activation='relu')(model2)
model2 = MaxPooling2D(2, 2)(model2)
model2 = Conv2D(160, (3, 3), activation='relu')(model2)
model2 = MaxPooling2D(2, 2)(model2)
model2 = Flatten()(model2)

model = concatenate([model1, model2])

model = Dense(300, activation='relu')(model)
model = Dropout(dropout_rate)(model)
model = Dense(300, activation='relu')(model)
model = Dropout(dropout_rate)(model)
model = Dense(300, activation='relu')(model)
model = Dropout(dropout_rate)(model)
model = Dense(300, activation='relu')(model)
model = Dropout(dropout_rate)(model)
model = Dense(300, activation='relu')(model)
model = Dense(7, activation='softmax')(model)

model = Model(inputs=inputs, outputs=model)
```

第六個與第七個使用了 ResNet 架構，其中有個 stack function 長這樣

```
def stack(x, h):
    conv = Conv2D(h // 4, (1, 1), activation='relu', padding='same')(x)
    conv = Conv2D(h // 4, (3, 3), activation='relu', padding='same')(conv)
    conv = Conv2D(h, (1, 1), activation='relu', padding='same')(conv)
    result = add([x, conv])
    return Activation('relu')(result)
```

第六個

```
dropout_rate = 0.15

model = Conv2D(64, (3, 3), activation='relu', padding='same')(inputs)
model = stack(model, 64)
model = MaxPooling2D((2, 2), padding='same')(model)
model = Dropout(dropout_rate)(model)
model = Conv2D(128, (3, 3), activation='relu', padding='same')(model)
model = stack(model, 128)
model = MaxPooling2D((2, 2), padding='same')(model)
model = Dropout(dropout_rate)(model)
model = Conv2D(256, (3, 3), activation='relu', padding='same')(model)
model = stack(model, 256)
model = MaxPooling2D((2, 2), padding='same')(model)
model = Dropout(dropout_rate)(model)
model = Conv2D(512, (3, 3), activation='relu', padding='same')(model)
model = AveragePooling2D((2, 2), padding='same')(model)
model = Dropout(dropout_rate)(model)

model = Flatten()(model)
model = Dense(400, activation='relu')(model)
model = Dense(7, activation='softmax')(model)

model = Model(inputs=inputs, outputs=model)
```

第七個

```
dropout_rate = 0.15

model = Conv2D(64, (3, 3), activation='relu', padding='same')(inputs)
model = stack(model, 64)
model = MaxPooling2D((2, 2), padding='same')(model)
model = Dropout(dropout_rate)(model)
model = Conv2D(128, (3, 3), activation='relu', padding='same')(model)
model = stack(model, 128)
model = stack(model, 128)
model = MaxPooling2D((2, 2), padding='same')(model)
model = Dropout(dropout_rate)(model)
model = Conv2D(256, (3, 3), activation='relu', padding='same')(model)
model = stack(model, 256)
model = stack(model, 256)
model = stack(model, 256)
model = MaxPooling2D((2, 2), padding='same')(model)
model = Dropout(dropout_rate)(model)
model = Conv2D(512, (3, 3), activation='relu', padding='same')(model)
model = AveragePooling2D((2, 2), padding='same')(model)
model = Dropout(dropout_rate)(model)

model = Flatten()(model)
model = Dense(512, activation='relu')(model)
model = Dense(256, activation='relu')(model)
model = Dense(128, activation='relu')(model)
model = Dense(7, activation='softmax')(model)

model = Model(inputs=inputs, outputs=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？

   我的 normalization 是分別對每張圖片做標準化，就是計算其平均值與標準差之後，將每個像素扣掉標準差然後除以平均。在都有使用 data augmentation 還有其他參數都一樣的情況下跑一樣的 epoch：

   沒做 normalization: 0.64084

   有做 normalization: 0.64837

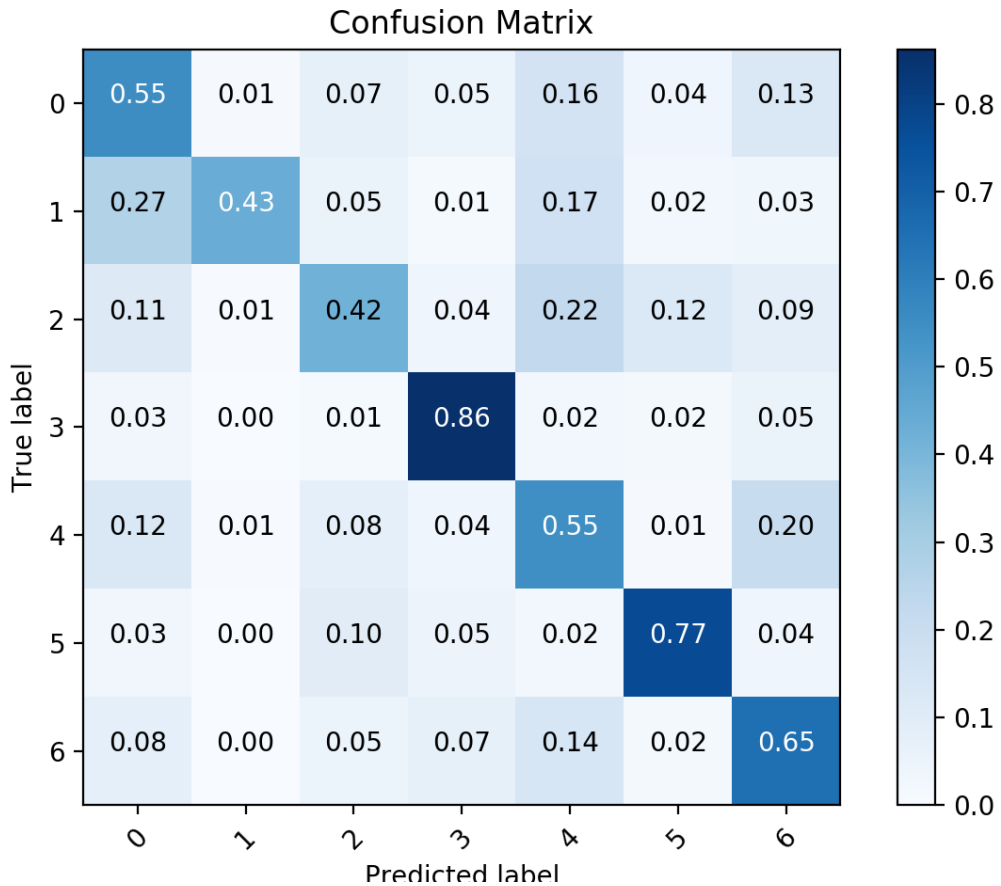
   我的 augmentation 是使用 keras 中的 ImageDataGenerator，相關數據如下圖

```
ImageDataGenerator(    featurewise_center=False,
                       samplewise_center=False,
                       rotation_range=15,
                       width_shift_range=0.3,
                       shear_range=0.3,
                       height_shift_range=0.3,
                       zoom_range=0.3,
                       data_format='channels_last')
```

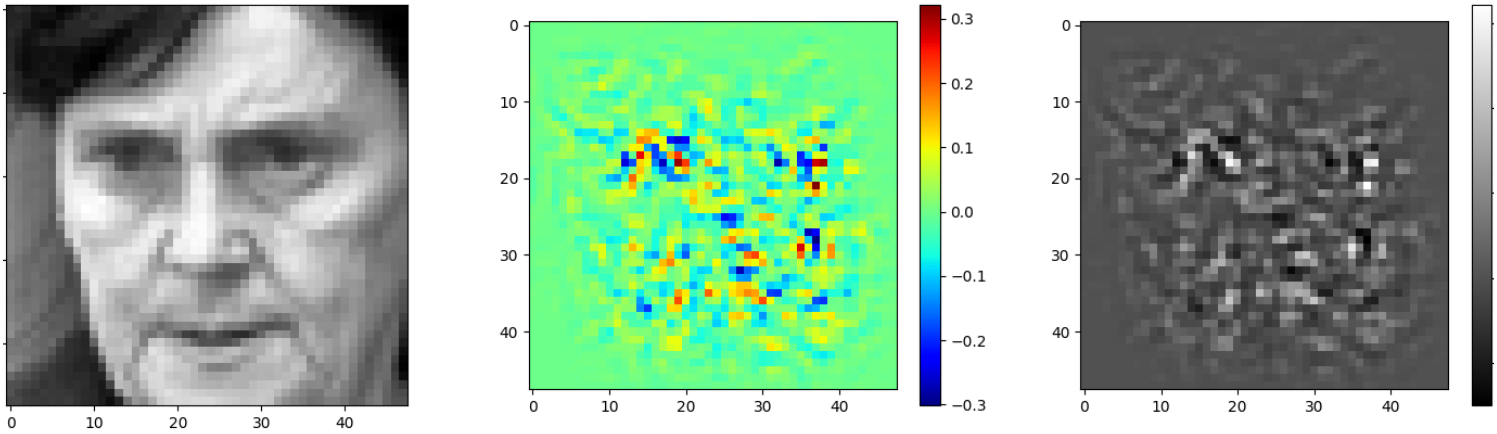
   有做 augmentation: 0.64084

   沒做 augmentation: 0.58010

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]

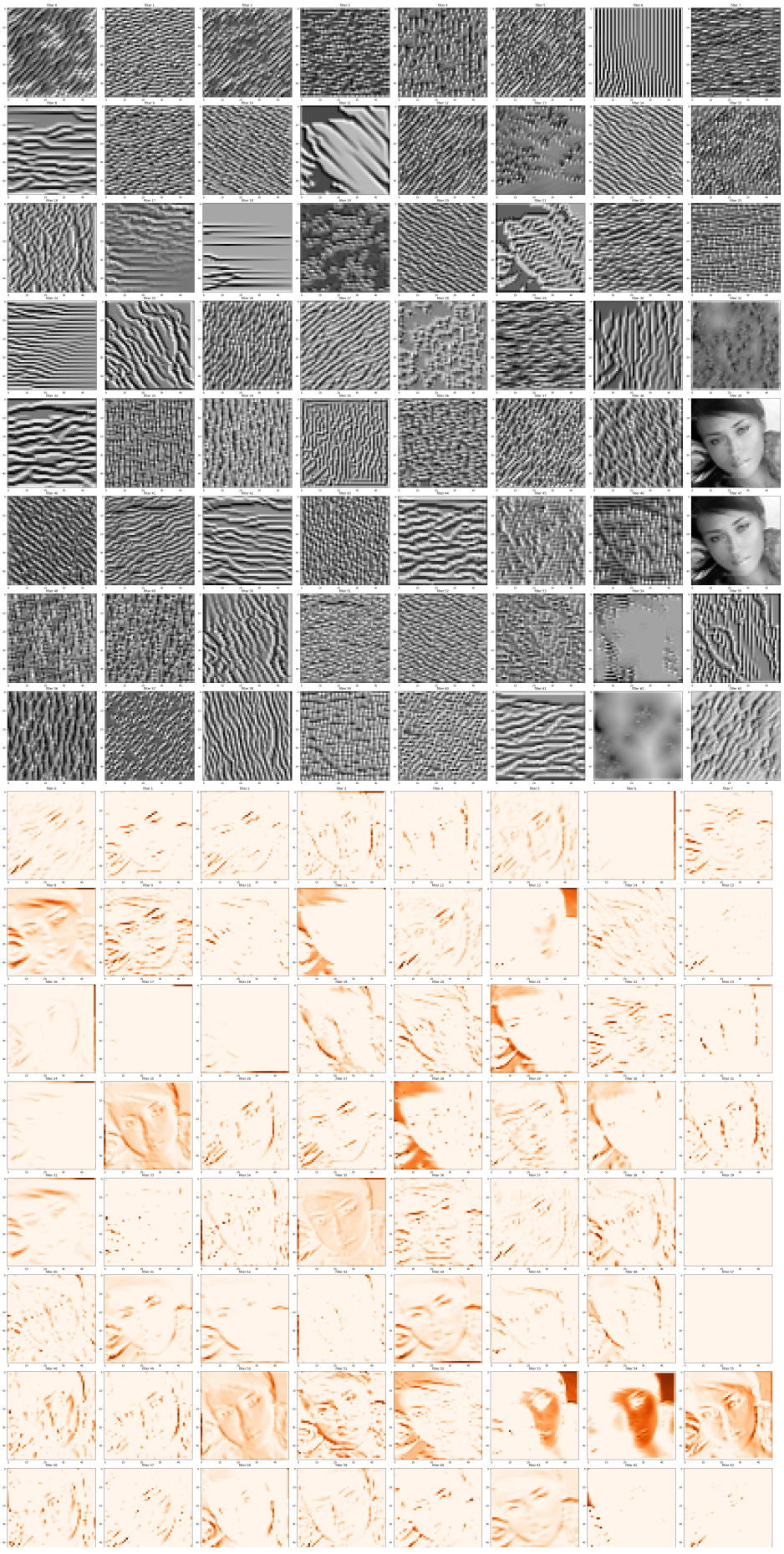


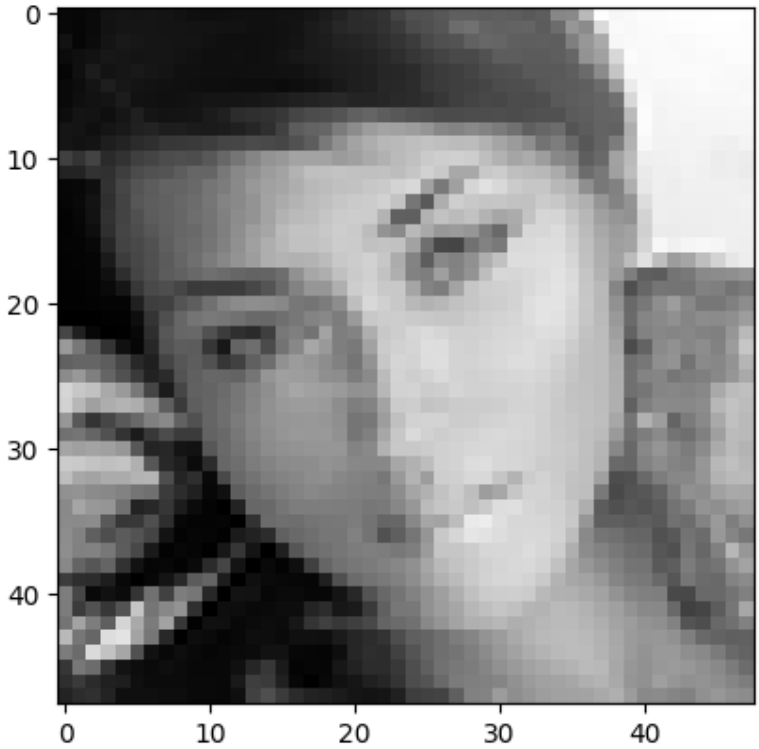
4. (1%) 從(1)(2)可以發現，使用 CNN 的確有些好處，試繪出其 saliency maps，觀察模型在做 classification 時，是 focus 在圖片的哪些部份？



從 saliency maps 看起來，這個模型比較著重在於眼睛、鼻子、以及臉部輪廓。我原本預期嘴巴會佔絕大部份，但是這樣看來嘴巴的影響並不太大，倒是鼻子跟嘴巴中間還比較重要。

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。





我使用的 layer 是我第一題的第七個 model 中的 add_1，圖中可以看到 64 個 filter 之中有兩個是原圖的形式，而通過這個 layer 大部分的 filter 的圖都大幅保留了原圖的輪廓，有的針對臉的形狀，而有的比較凸顯五官的部分。然而有些 filter 的 output 是一片白，我想是這些 filter 與圖之相關性極低的關係，也許在 training 中對這些 filter 做取捨可以對預測的準確性有更多的提升。