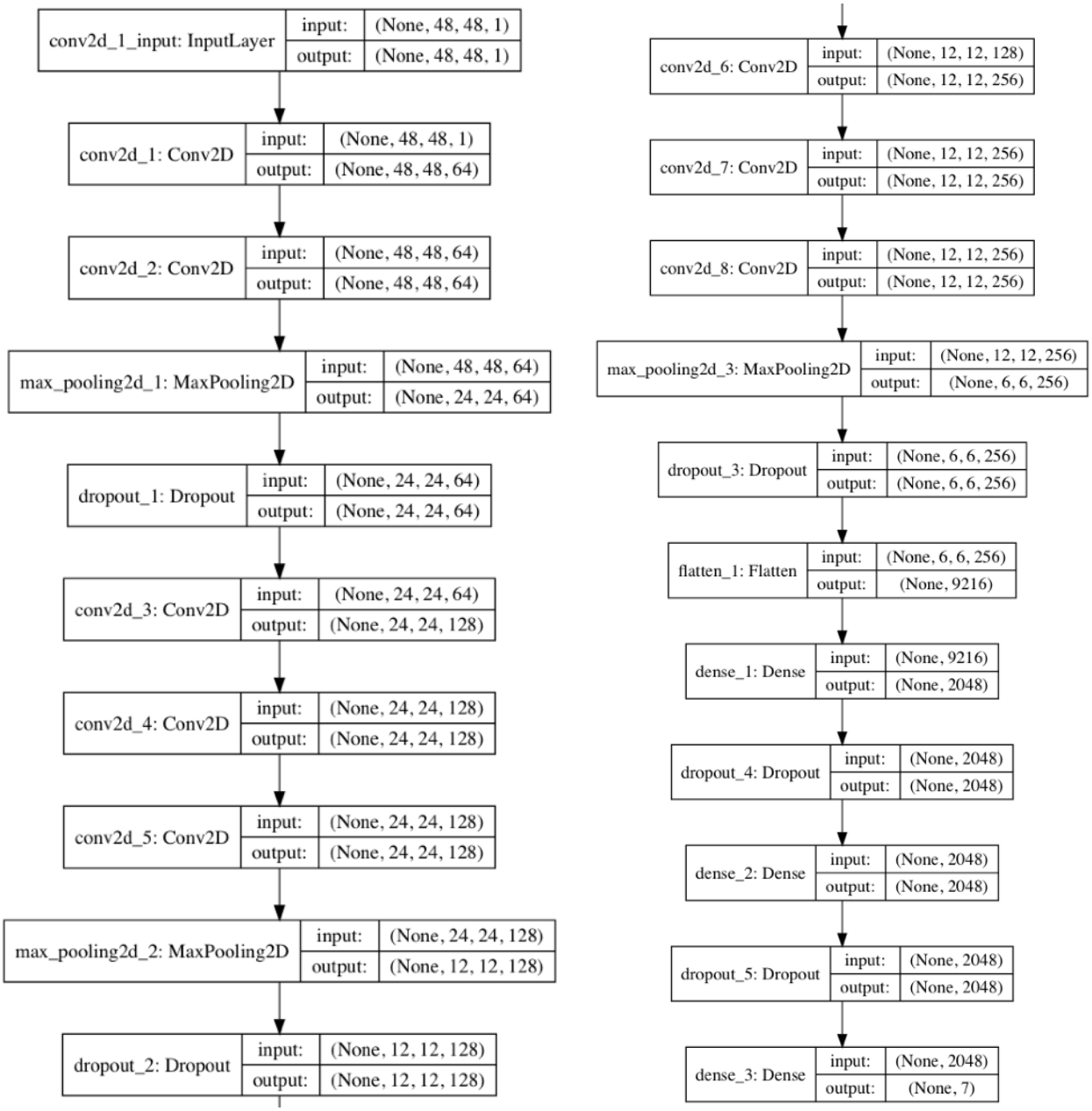
學號：B03901039 系級： 電機三 姓名：童寬

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：

(1) 模型架構 (左圖的底部接到右圖的開頭)



模型參數：24,969,031 個

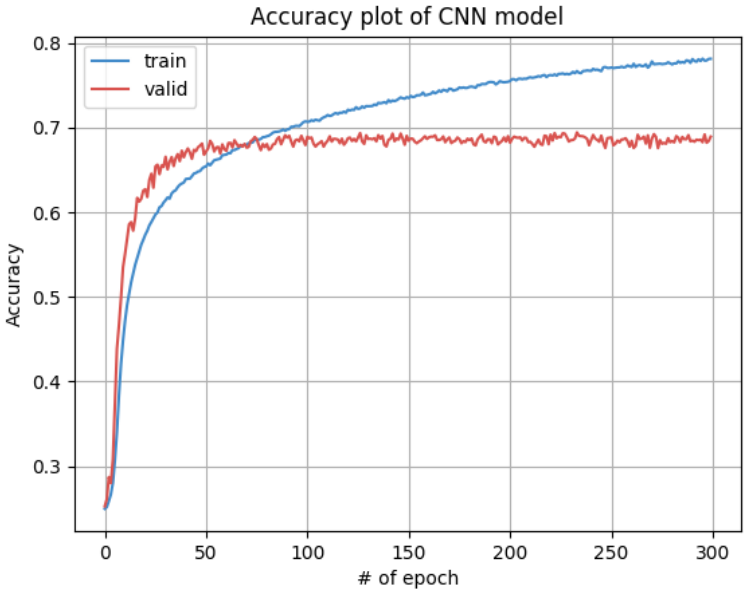
Dropout rate：0.5

使用 ImageGenerator 產生新的訓練資料 (rotation, width shift, height shift, zoom, horizontal flip)

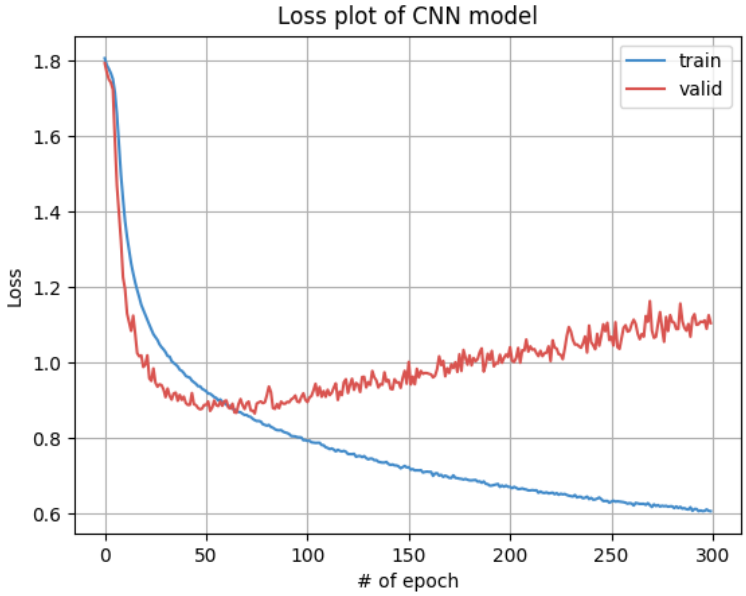
- 原本打算按照 vgg19 的模型去實作，但實驗結果發現 19 層的網路在這個問題，太容易 overfitting，因此後來改成稍微少層一點。
- 上面的 model 是我實做出來，能在 validation data 上取得最好結果的 model。

學號：B03901039 系級： 電機三 姓名：童寬

(2) 訓練過程



- Training accuracy 還可以再提高
- Validating accuracy 已經開始停滯了
- 一開始因為有 dropout 的關係，validating 的表現會比 training 好



- Training loss 還能再下降
- Validating loss 在第 50 個 epoch 開始往上升
- 一開始因為有 dropout 的關係，validating 的表現會比 training 好

(3) 準確率

Training ：0.78156150753818476 (#epoch = 295)

Validating ：0.69440000133514401 (#epoch = 233)

Testing ：0.68181 (#epoch = 233)

(Training 跟 Validating 都是選擇最好的 model，Testing 則是根據最好的 Validating model 丟到 Kaggle 去測試)

(4) 結果討論

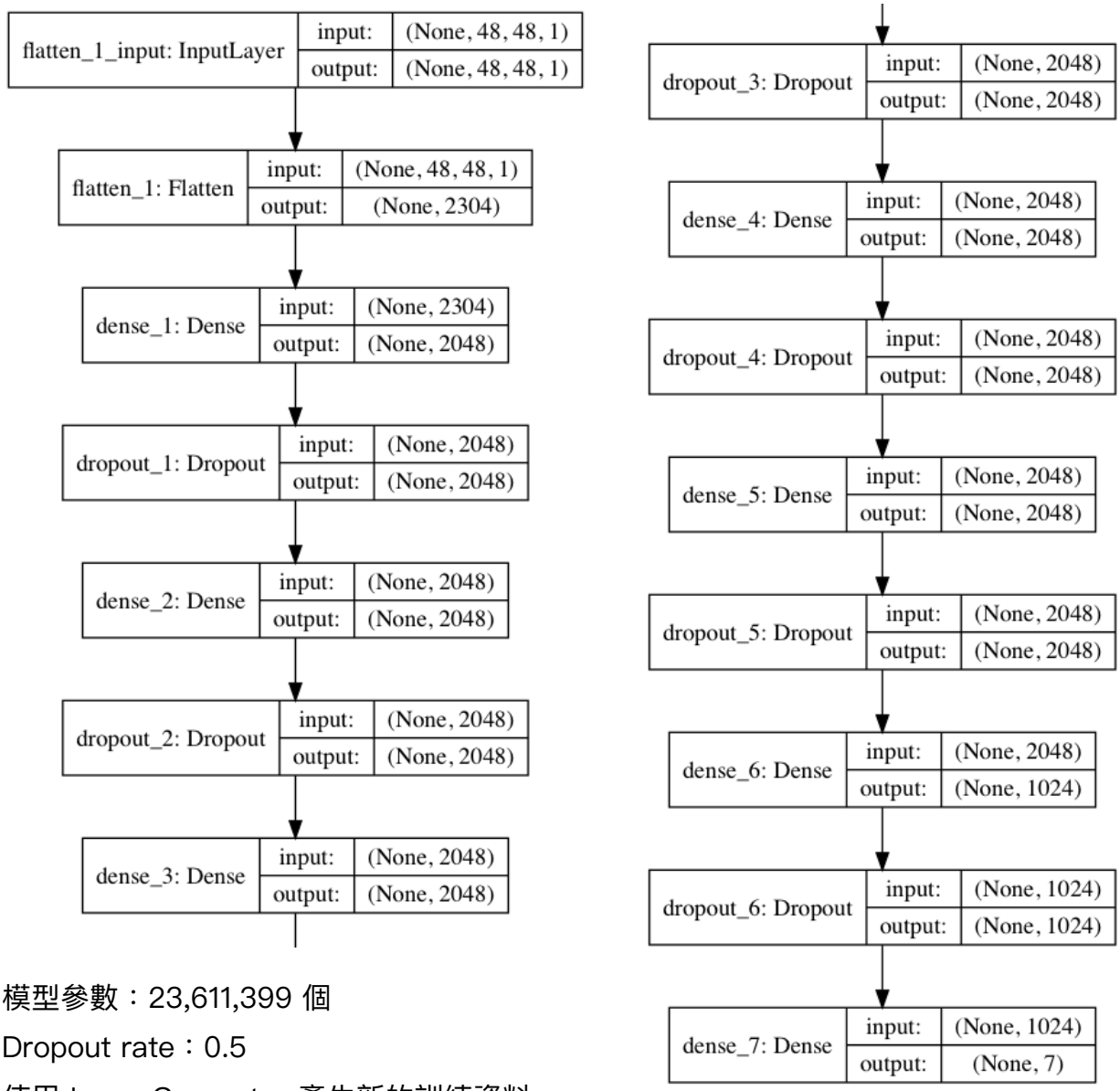
- 雖然 Training accuracy 跟 loss 都還能再進步，但 Validating loss 已經開始往上升，代表 model 已經開始 overfitting 了

學號：B03901039 系級： 電機三 姓名：童寬

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：

(1) 模型架構 (左圖的底部接到右圖的開頭)



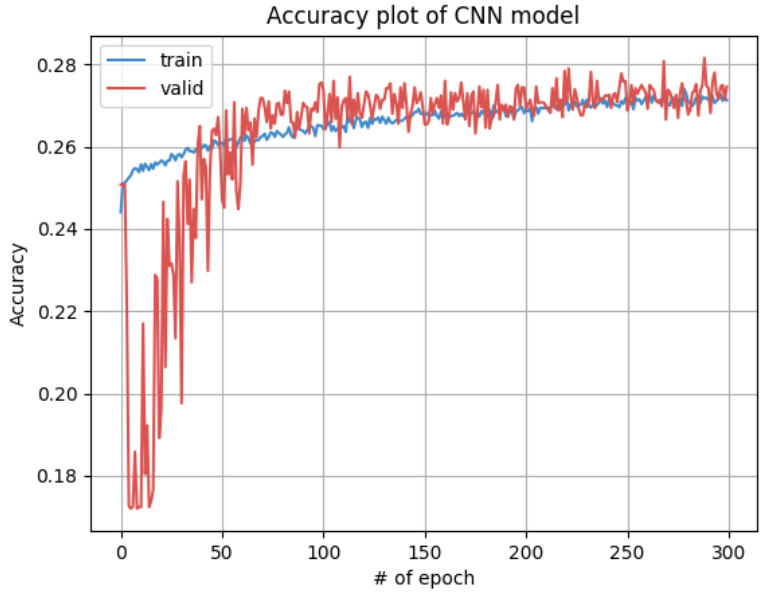
模型參數：23,611,399 個

Dropout rate：0.5

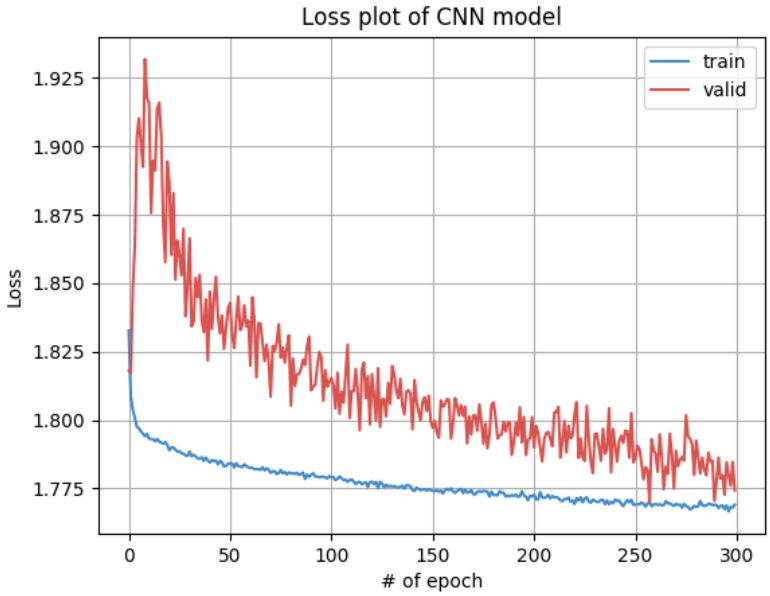
使用 ImageGenerator 產生新的訓練資料

(rotation, width shift, height shift, zoom, horizontal flip)

(2) 訓練過程



- Training accuracy 因為有太多參數，所以在跟 CNN 相同的 epoch 做比較，根本沒有太大的進步
- Validating accuracy 的跳動很大，應該是因為這個 model 目前還處在亂猜的狀態



- Training loss 一樣因為太多參數，loss 下降速度跟 CNN 比起來，慢了非常多
- Validating loss 一樣跳動非常大

(3) 準確率

Training ：0.27401378273939603 (#epoch = 278)

Validating ：0.28160000061988832 (#epoch = 288)

Testing ：0.27250 (#epoch = 288)

(Training 跟 Validating 都是選擇最好的 model，Testing 則是根據最好的 Validating model 丟到 Kaggle 去測試)

(4) 結果討論

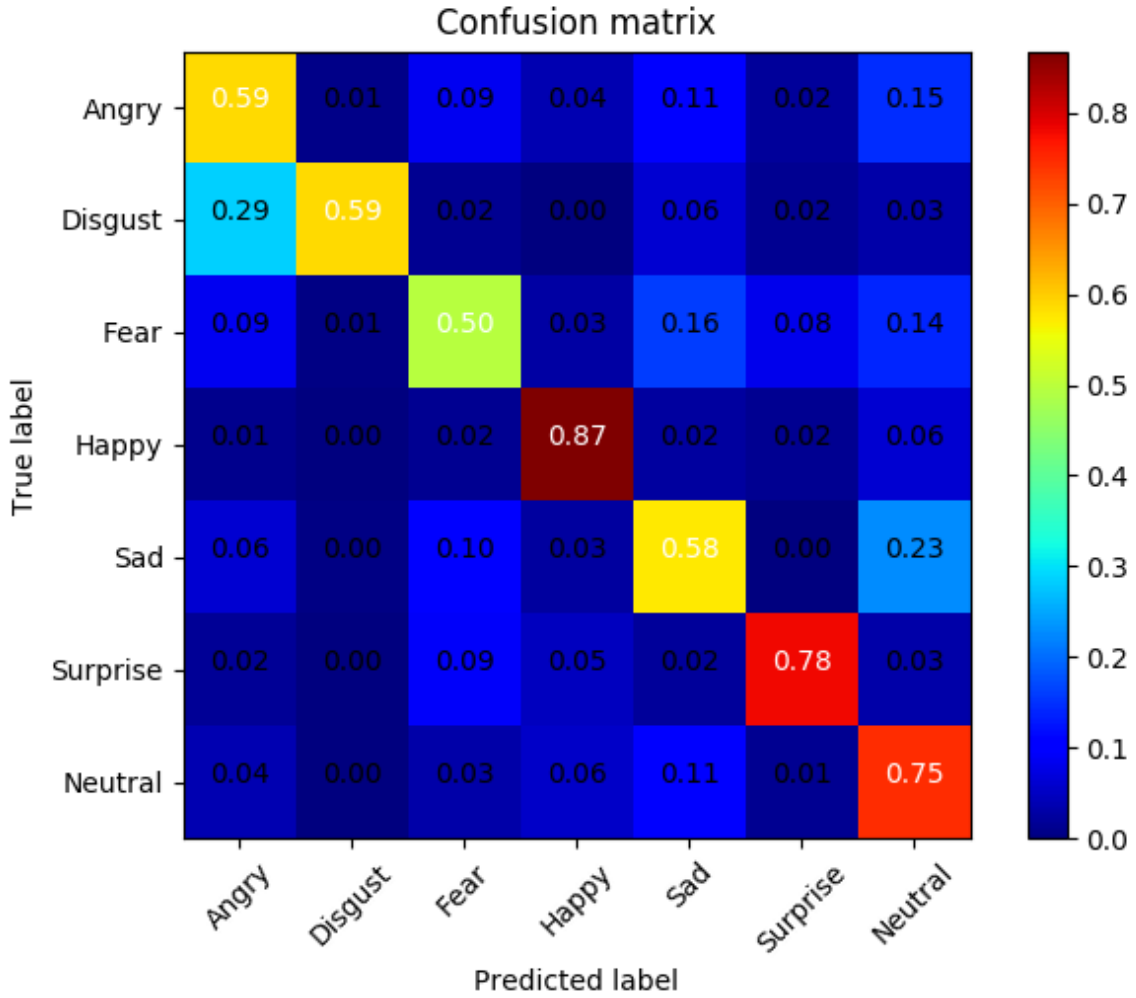
- 跟 CNN 一樣參數的 DNN，因為參數過多，又沒有像 CNN 去調整 model 的結構，使得其表現在 Training set 上就已經很差了，沒有辦法訓練起來。

學號：B03901039 系級： 電機三 姓名：童寬

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：

(1) Confusion matrix



- Angry：正確率 59%，容易跟 Neutral (15%), Sad (11%), Fear (9%) 用混
- Disgust：正確率 59%，容易跟 Angry (29%) 用混
- Fear：正確率 50%，容易跟 Sad (16%), Neutral (14%) 用混
- Happy：正確率 87%，容易跟 Neutral (6%) 用混
- Sad：正確率 58%，容易跟 Neutral (23%), Fear (10%) 用混
- Surprise：正確率 78%，容易跟 Fear (9%) 用混
- Neutral：正確率 75%，容易跟 Sad (11%) 用混

(2) 結果討論

- 有一個奇怪的地方，當 True label 是 Angry 時，幾乎不太會猜是 Disgust。但當 True label 是 Disgust 時，竟然有 23% 會去猜 Angry。
- Neutral 很容易跟其他用混，我猜想是因為 Neutral 這個表情的定義沒有其它那麼清楚。

學號：B03901039 系級： 電機三 姓名：童寬

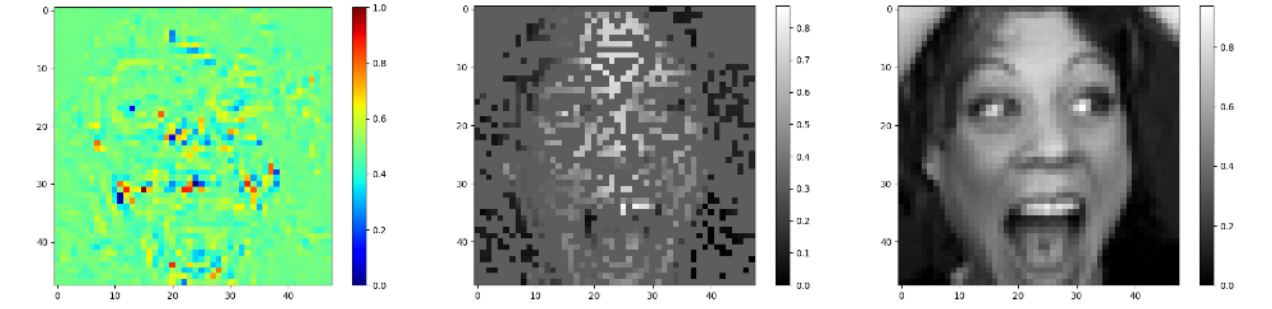
4. (1%) 從(1)(2)可以發現，使用 CNN 的確有些好處，試繪出其 saliency maps，觀察模型在做 classification 時，是 focus 在圖片的哪些部份？

答：

(1) Saliency maps

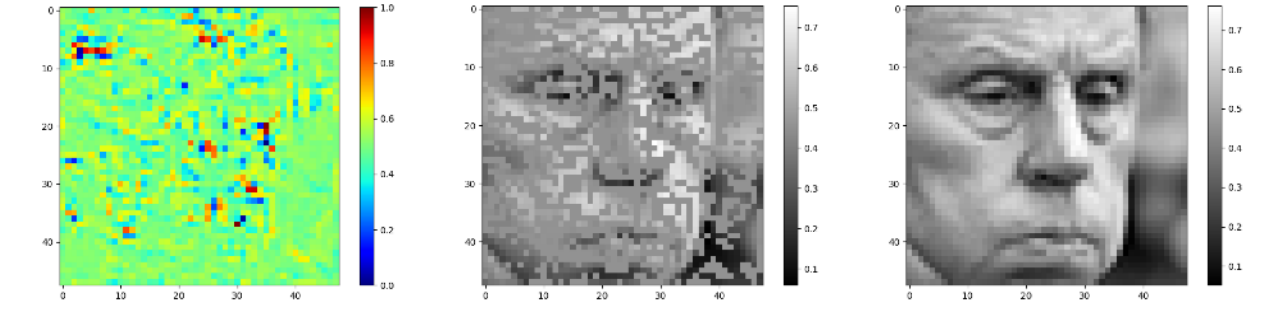
我先預測 testing set 每張圖片屬於什麼表情，再連同特徵，算出 gradient，以下是結果：

生氣



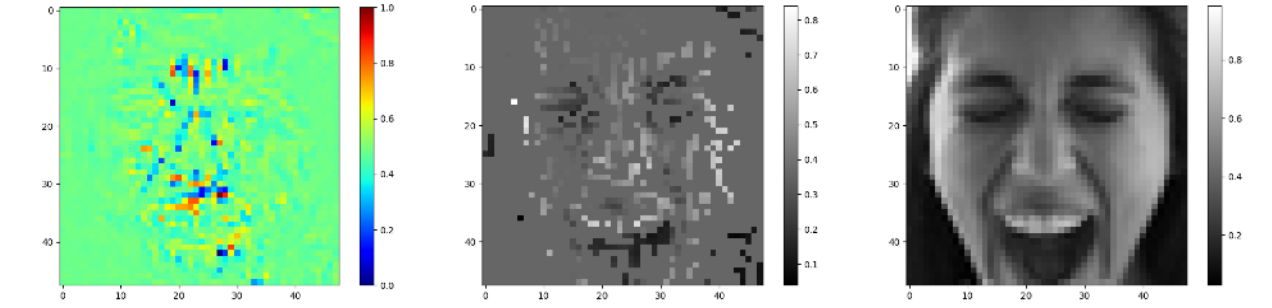
可以看出模型有特別注重張大的眼睛跟嘴巴。

厭惡



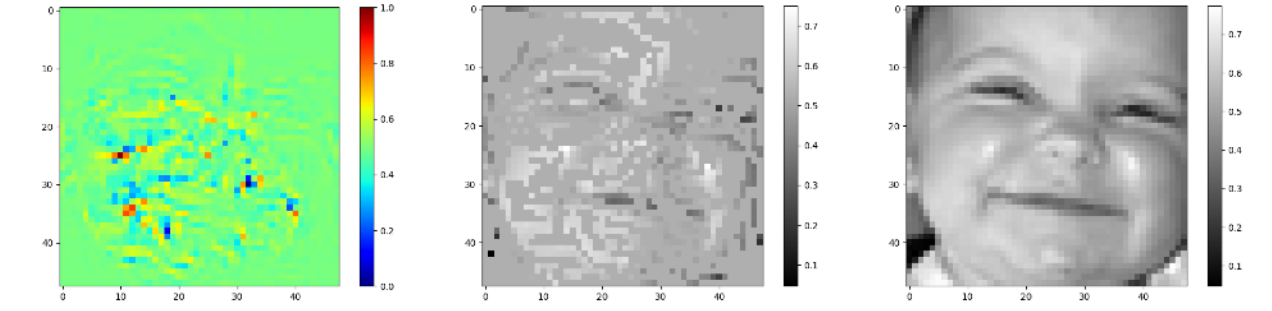
我覺得這張模型有找到往下彎的嘴巴，還有眼睛周圍的紋路。

恐懼



模型有看到閉起來的眼睛，跟露出牙齒，張大的嘴巴。

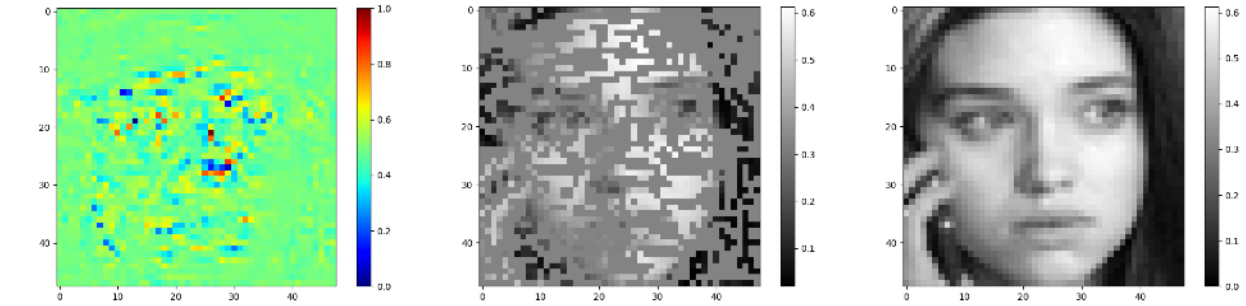
高興



學號：B03901039 系級： 電機三 姓名：童寬

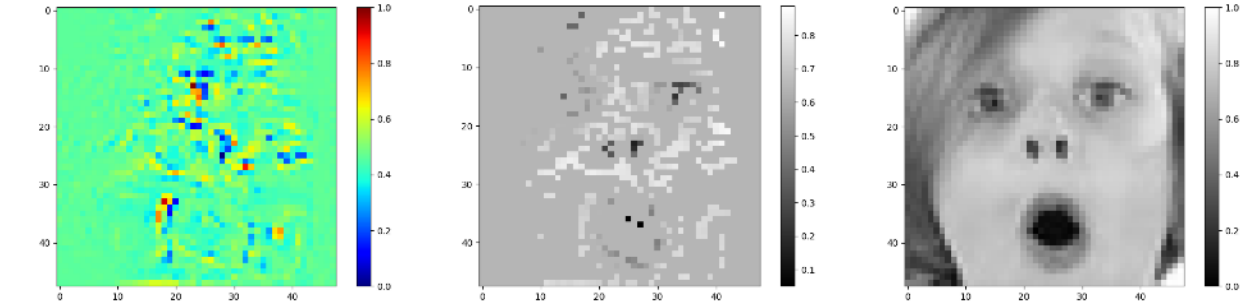
瞇起來的眼睛跟微笑的嘴巴是模型專注的地方。

難過



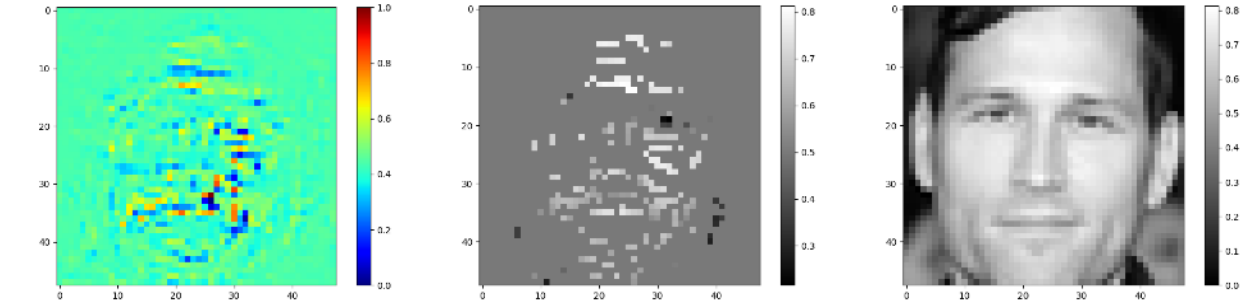
模型專注的似乎是眼睛的表情。

驚訝



可以看出模型看到了張大而且是圓形的嘴巴，還有撐大的鼻孔。

中立



模型看到了沒有特別角度的嘴巴跟右眼。

(2) 結果討論

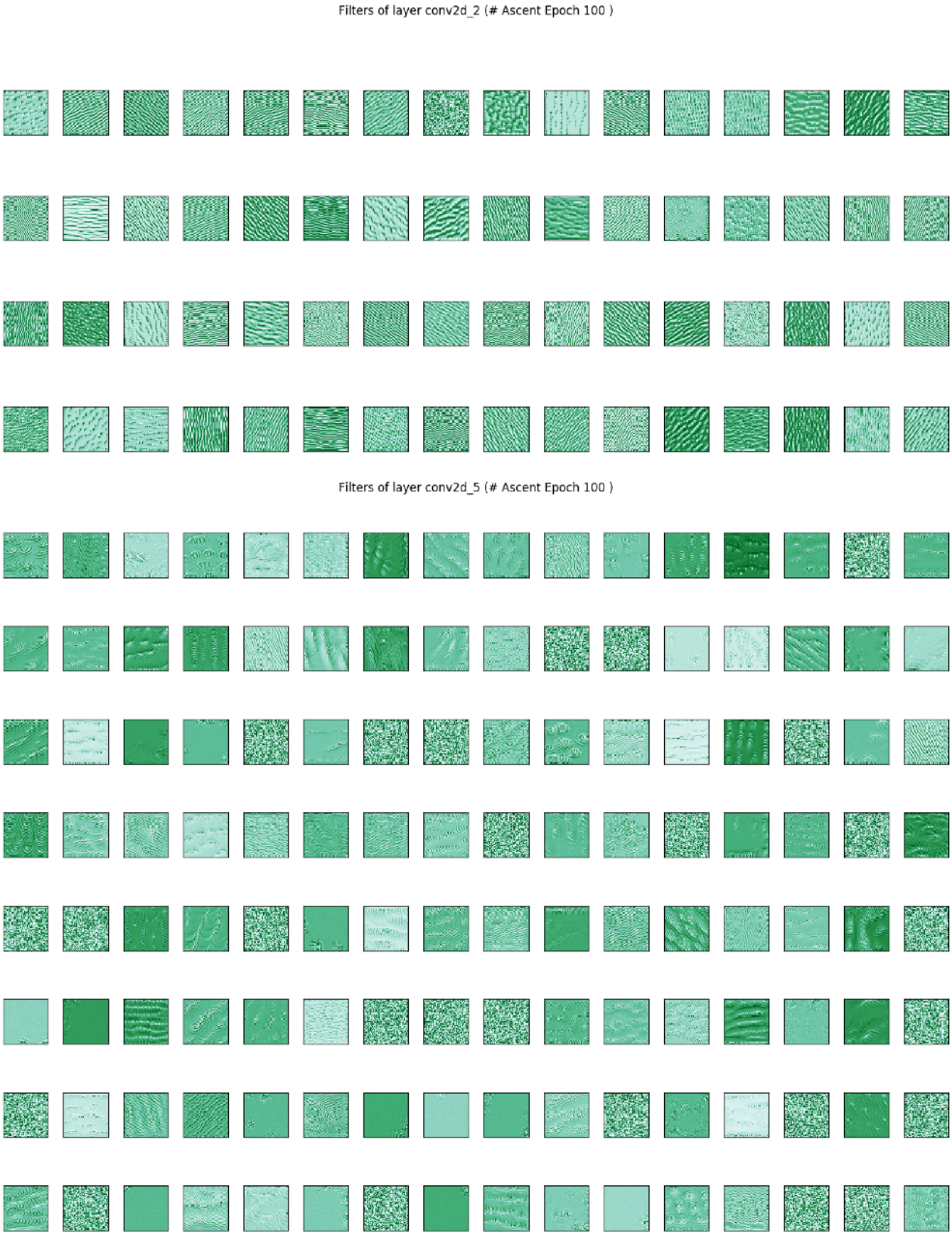
- 並不是每一張圖的結果都能看出結果，有些圖大部分的地方 gradient 都很高
- 我本來以為找不到專注點的，才會被分到中立，但沒有想中立的表情也是有特徵的，但他的特徵就是平淡的表情XD

學號：B03901039 系級： 電機三 姓名：童寬

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：

(1) Gradient Ascent 找出最能 activate filter 的影像 (選擇前中兩層 convolution layer)

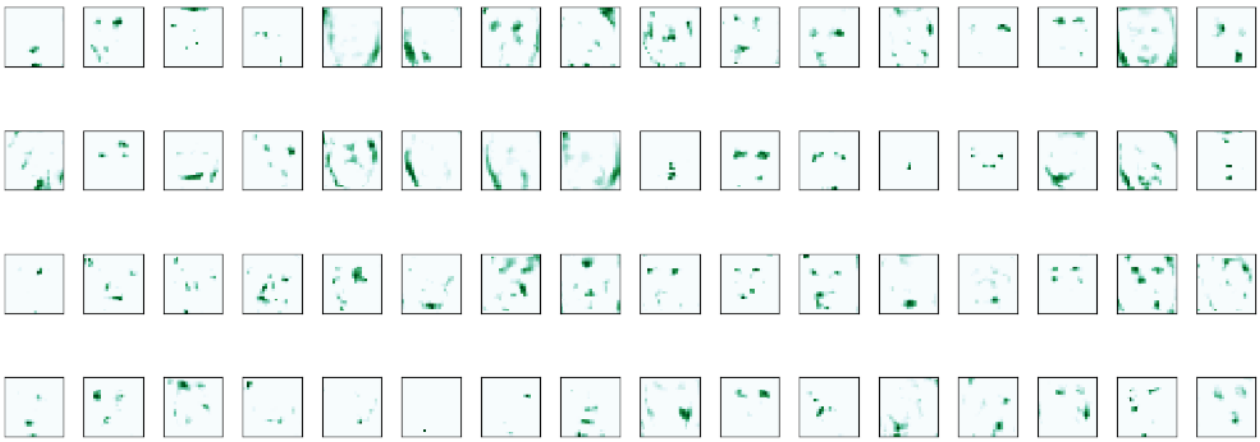


學號：B03901039 系級： 電機三 姓名：童寬

(2) 輸入影像，觀察其在 layer 的 output (選擇前中兩層 convolution layer) (conv2d_5 只列前 64 個 filter)

Output of layer conv2d_2 (Given image74)

Output of layer conv2d_5 (Given image74)



(3) 結果討論

- 第一小題可以看出，一開始 filter 注重的是簡單、不同角度的線條。值得注意的是有滿多 filter 注重的地方都很像，這或許代表一開始根本不需要這麼多的 filter。
- 在第一小題的 conv2d_5 這張圖中，可以看到 filter 注重的是一些圖案，例如：眼睛、鼻孔、嘴巴等重複的五官。代表它們真的有抓到人臉的特徵。
- 第二小題可以看出，filter 可以抓到人臉的線條，五官的位置，代表 filter 關注的地方真的是它應該關注的地方，而不是以無法解釋的方法再做判斷。
- 在第二小題的 conv2d_5 這張圖中，雖然因為經過了一次 maxpooling，使得圖片少了一半的像素，但還是可以看出 filter 注重的一樣是五官所在的位置。

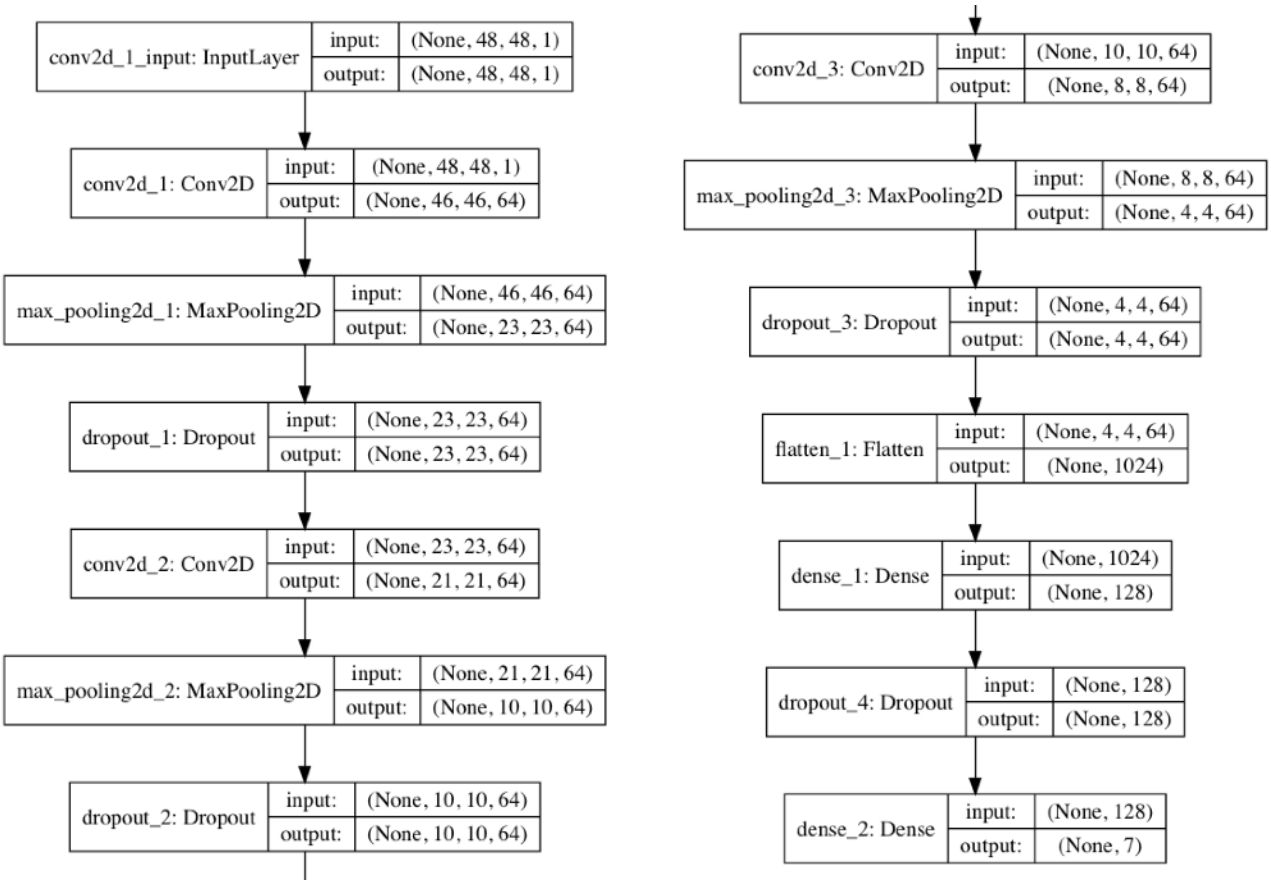
學號：B03901039 系級： 電機三 姓名：童寬

[Bonus] (1%) 從 training data 中移除部份 label，實做 semi-supervised learning

(1) 實作方法

1. 將 20000 筆 training data 分出來當 unlabel data
2. 以剩餘的 training data 進行訓練，epoch 設為 10
3. Predict unlabel data，將 entropy 前 2000 小的加到 training data
4. 以新的 training data 進行訓練，epoch 設為 10，用 entropy-based regularization 作為 loss function
5. 重複 3~4 步驟直到全部的 unlabel data 都加到 training data 上
6. 以全部的 training data 進行最後的訓練，epoch 設為 40

(2) 模型架構 (左圖的底部接到右圖的開頭)



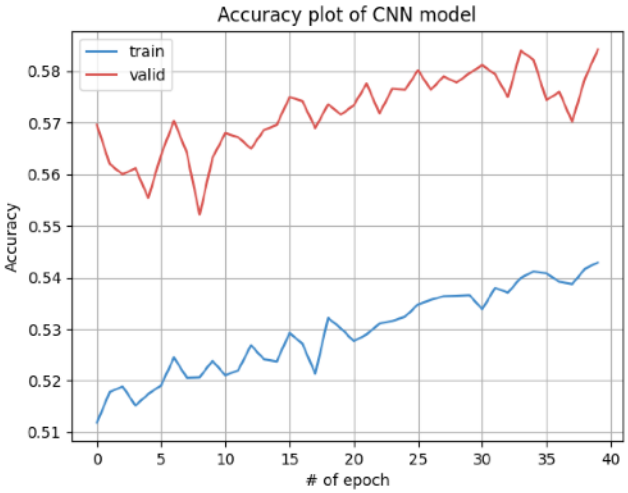
模型參數：206,599 個

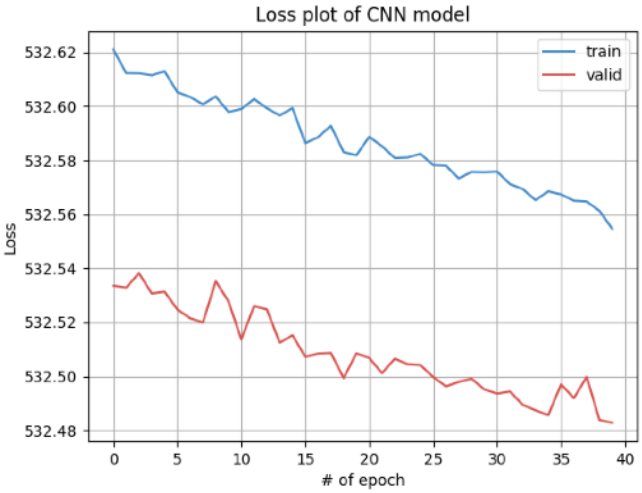
Dropout rate：0.5

沒有使用 ImageGenerator 產生新的訓練資料

學號：B03901039 系級： 電機三 姓名：童寬

(3) 訓練過程 (記錄最後 40 個 epoch)





- valiging 的準確率一直都比 training 高，模型適應未知資料的能力有所提升。

(4) 準確率

Training ：0.54291619191694063 (#epoch = 39)

Validating ：0.58420000209808354 (#epoch = 39)

Testing ：0.56367 (#epoch = 39)

(Training 跟 Validating 都是選擇最好的 model，Testing 則是根據最好的 Validating model 丟到 Kaggle 去測試)

(5) 結果討論

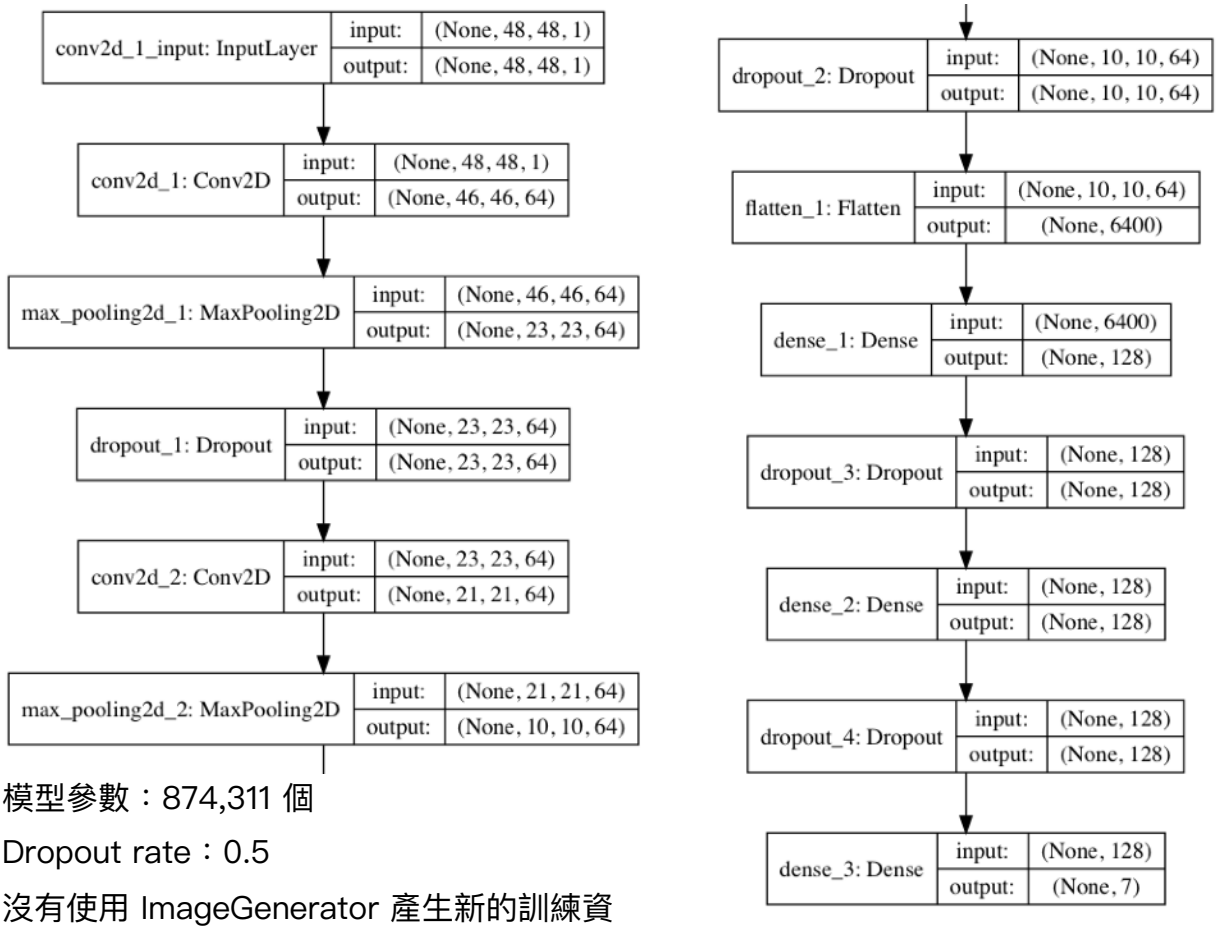
- 從訓練過程可以看出，semi-supervised 能有效防止 overfitting 的發生，或許我可以嘗試在更複雜的模型上加入此方法。

學號：B03901039 系級： 電機三 姓名：童寬

[Bonus] (1%) 在Problem 5 中，提供了3個 hint，可以嘗試實作及觀察 (但也可以不限於 hint 所提到的方向，也可以自己去研究更多關於 CNN 細節的資料)，並說明你做了些什麼？[完成1個: +0.4%, 完成2個: +0.7%, 完成3個: +1%]

(1) 用表現較差的 model 觀察 filter

i. 模型架構



模型參數：874,311 個

Dropout rate：0.5

沒有使用 ImageGenerator 產生新的訓練資料

ii. 準確率

Training ：0.77396769187296455 (#epoch = 297)

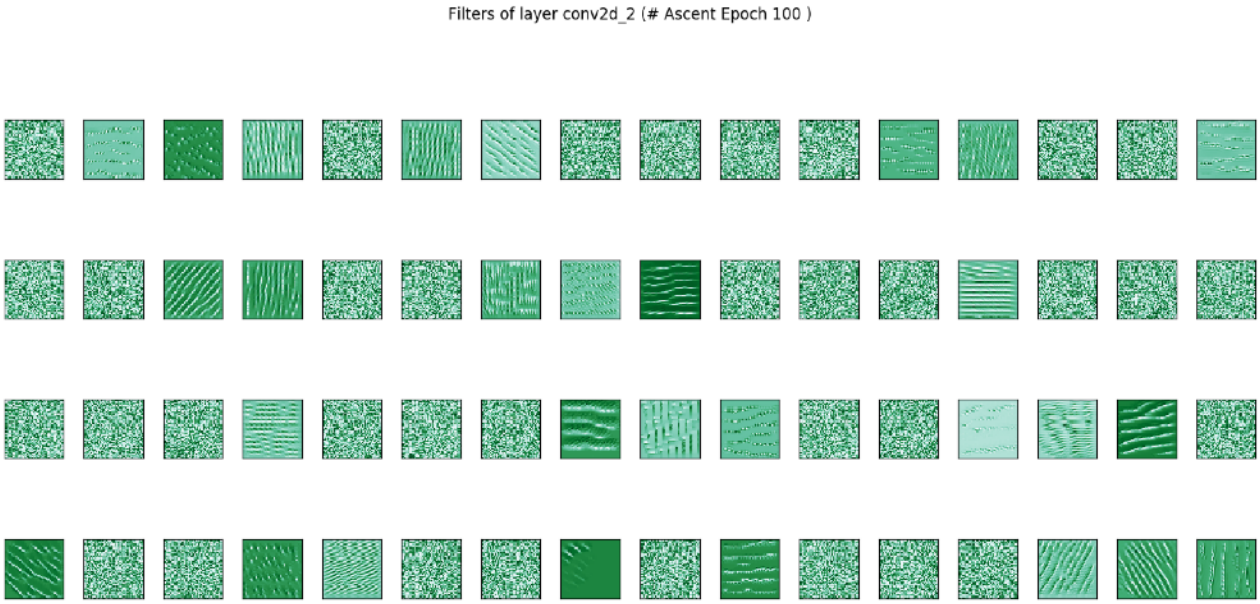
Validating ：0.59760000028610227 (#epoch = 112)

Testing ：0.57593 (#epoch = 112)

(Training 跟 Validating 都是選擇最好的 model，Testing 則是根據最好的 Validating model 丟到 Kaggle 去測試)

學號：B03901039 系級： 電機三 姓名：童寬

iii. Gradient Ascent 找出最能 activate filter 的影像



跟第五題的結果相比，我覺得有以下幾個不同的地方

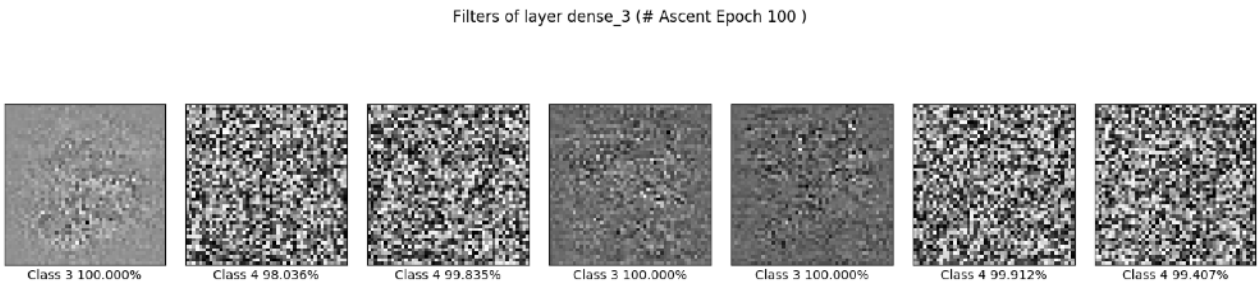
- 像是亂碼的 filter 變多
- 辨認特徵的 filter 種類變少
- 出現更多重複、辨認類似 pattern 的 filter

iv. 輸入影像，觀察其在 layer 的 output

layer output 的差別沒有這麼多，一樣有看到 filter 會專注在五官跟臉的位置。

學號：B03901039 系級： 電機三 姓名：童寬

(2) 找出最能 activate 特定 class 的影像



圖片下方的敘述代表該圖最能 activate 的 class，旁邊的數字是屬於該 class 的 softmax 值取百分比。

遇到的問題

雖然我有指定要對哪一個神經元做 gradient ascent，但跑出來的結果卻不是每個 class 都有。