學號：R07941023 系級： 光電碩二 姓名：呂彥穎

1. (1%) 請說明這次使用的 model 架構，包含各層維度及連接方式。

Sol:

要用當然用我查到最屌的-model:densenet201-改

(這裡 in/out 是指 function 吃進去的維度, 不代表 feature 總數量)

Image size = (3, 48, 48)

Initialization:

1. Convolution (out=(64, 24, 24), n=64, kernel=(7, 7), stride=(2, 2), padding=(3, 3))
2. BatchNorm (out=(64, 24, 24),)
3. ReLU (out=(64, 24, 24),)
4. Pooling(out=(64, 12, 12),, kernel =3, stride=2, padding=1)

Layer: (維度為其第一層的範例)

1.BatchNorm (out=(64, 12, 12))
2.ReLU (out=(64, 12, 12))
3.Convolution (out=(128, 12, 12), n=128, kernel =(1, 1), stride=(1, 1))
4.BatchNorm (out=(128, 12, 12))
5.ReLU (out=(128, 12, 12))
6.Convolution (out=(32, 12, 12), kernel =(3, 3), stride=(1, 1), padding=(1, 1))

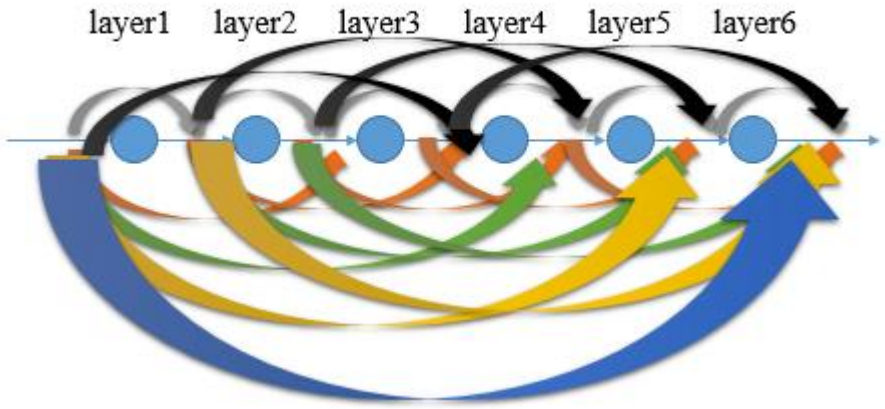
Transition: (維度為其第一次的範例)

1.BatchNorm (out=(256, 12, 12))
2.ReLU (out=(256, 12, 12))
3.Convolution(out=(128, 12, 12), n=128)
4. Pooling(out=(128, 12, 12))

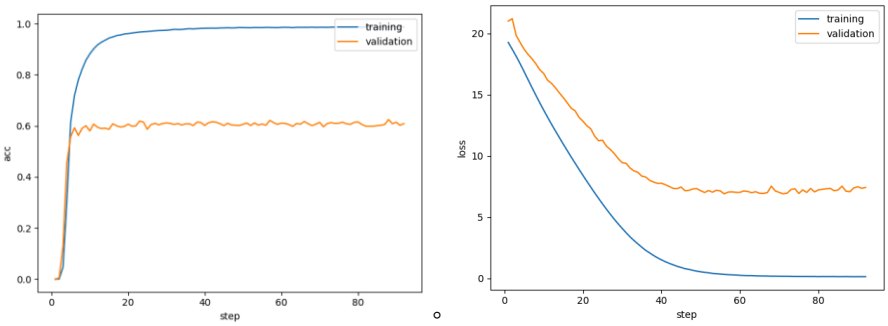
Initialization > block[6* Layer[> Transition > block[12* Layer] > Transition > block[48* Layer] > Transition > block[32* Layer] > BatchNorm (out=(1920, 1)) > linear (out=(1024, 1)) > ReLU > Dropout > linear (out=(512, 1)) > ReLU > Dropout > linear (out=(256, 1)) > ReLU > Dropout > linear (out=(7, 1))

其中每一個 block 內的連接方式除了前後連接還增加了 dense connection.(除了第 1 層, 每層 layer 的 input 皆會受到前面多個層的 output 值)

以第一個 block 為例:



2. (1%) 請附上 model 的 training/validation history (loss and accuracy)



3. (1%) 畫出 confusion matrix 分析哪些類別的圖片容易使 model 搞混，並簡單說明。
(ref: https://en.wikipedia.org/wiki/Confusion_matrix)

```
[[3783    4   69   43   81   11   50]
 [   8  421    7    3    6    1    1]
 [  63    2 3849   37   84   39   71]
 [  13    1   20 7169   24   22   35]
 [  68    4   63   40 4522    8  109]
 [  17    0   40   15    9 3070   21]
 [  40    0   37   63  101    9 4735]]
```

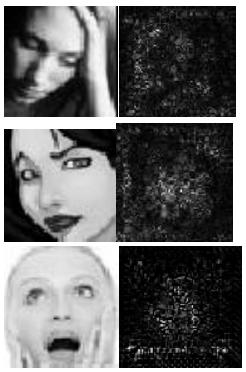
從 training+validation 整體的機率來看:

Class0 = 93.6%
Class1 = 94.1%
Class2 = 92.8%
Class3 = 98.4%
Class4 = 93.9%
Class5 = 96.7%
Class6 = 94.9%

以結果來看 Class0/ Class1/ Class2/ Class4/ Class6 均低於 95%, 容易出錯.

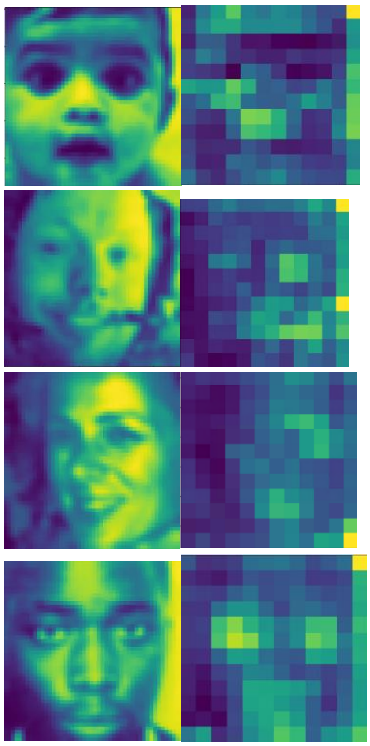
[關於第四及第五題]
可以使用簡單的 3-layer CNN model [64, 128, 512] 進行實作。

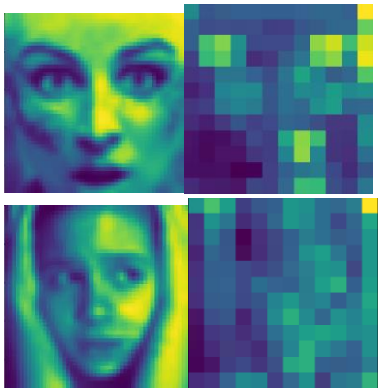
4. (1%) 畫出 CNN model 的 saliency map，並簡單討論其現象。
(ref: https://reurl.cc/Qpjg8b)



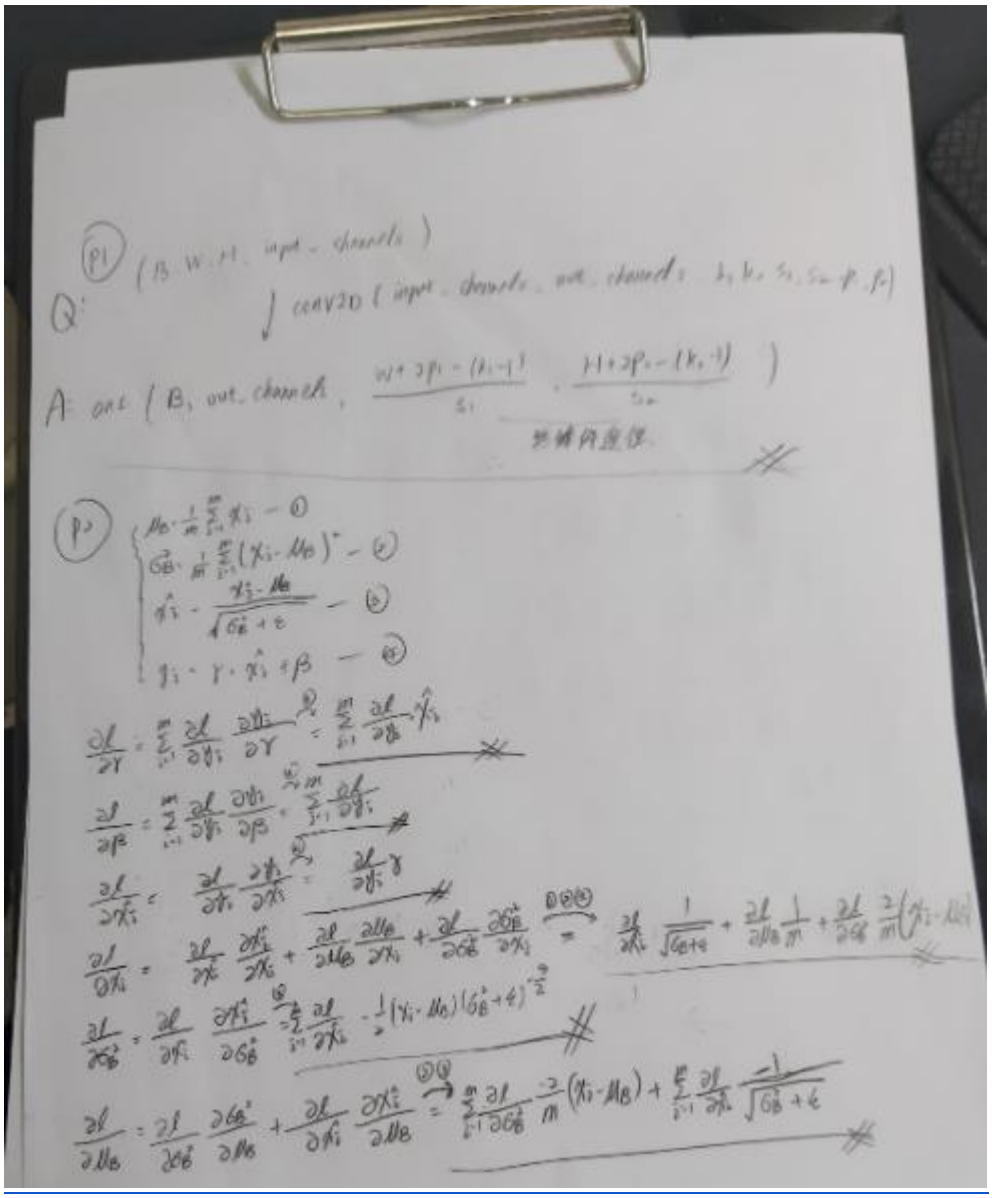
從結果來看, 因為 model 過於簡單了, 只能到整體形狀, 卻無法清楚的看見三個表情間的差異, 故準確率也不高.

5. (1%) 畫出最後一層的 filters 最容易被哪些 feature activate。
(ref: https://reurl.cc/ZnrgYg)





6. (3%)Refer to math problem
   https://hackmd.io/JIZ_0Q3dStSw0t0O0w6Ndw



(P3)

$\hat{y}_t = \frac{e^{z_t}}{\sum e^{z_i}}$ $\quad L_t(y_t, \hat{y}_t) = -y_t \log \hat{y}_t$ $\quad L(y, \hat{y}) = -\sum_i y_i \log \hat{y}_i$

$\frac{\partial L_t}{\partial z_t} = \begin{cases} \left(\sum_{i \neq t} \frac{\partial L_i}{\partial \hat{y}_i} \frac{\partial \hat{y}_i}{\partial z_t}\right) \\ + \frac{\partial L_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial z_t} \end{cases}$ $\quad \frac{\partial \hat{y}_t}{\partial z_t} = \frac{\partial}{\partial z_t}\left(\frac{e^{z_t}}{e^{z_t} + C}\right) = e^{z_t}(e^{z_t} + C)^{-1} - (e^{z_t} + C)^{-2} e^{z_t} e^{z_t}$

$= \hat{y}_t - \hat{y}_t^2$

$= \frac{\partial L_t}{\partial \hat{y}_t}(\hat{y}_t - \hat{y}_t^2)$, $\quad \frac{\partial L_t}{\partial \hat{y}_t} = \frac{\partial}{\partial \hat{y}_t}(-y_t \log \hat{y}_t)$

$= \begin{cases} \sum_{i \neq t} \frac{\partial L_i}{\partial \hat{y}_i} \frac{\partial \hat{y}_i}{\partial z_t} \\ + \end{cases}$

$= [y_t - y_t \hat{y}_t - \sum_{i \neq t} y_i \hat{y}_t]$

$= -[y_t(1 - \hat{y}_t) - \sum_{i \neq t} y_i \hat{y}_t]$

$= -[y_t - \hat{y}_t y_t - \hat{y}_t \sum_{i \neq k} y_i]$

$= -[y_t - \hat{y}_t(\sum_i y_i)]$

$= \hat{y}_t - y_t$