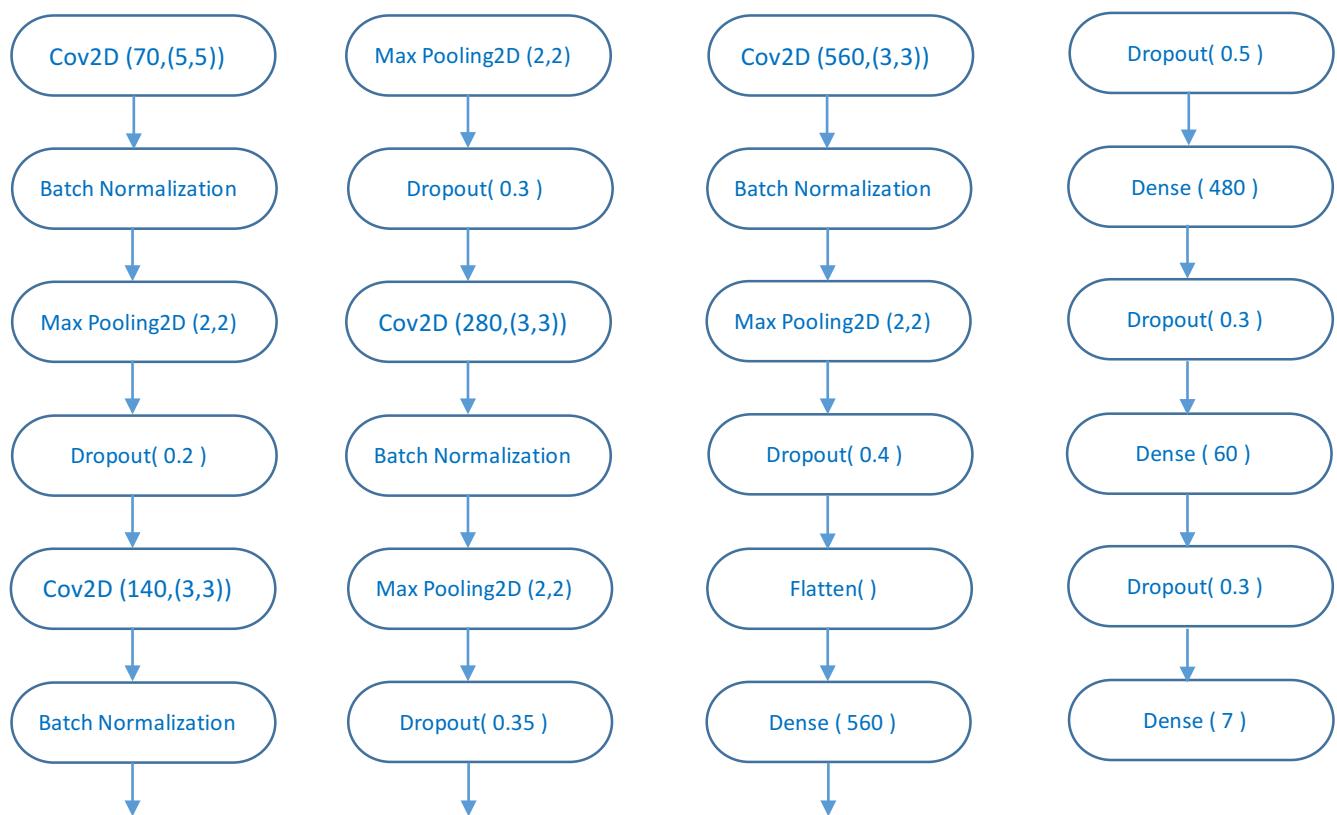


1. (1%) 請說明你實作的 CNN model，其模型架構、訓練參數和準確率為何？

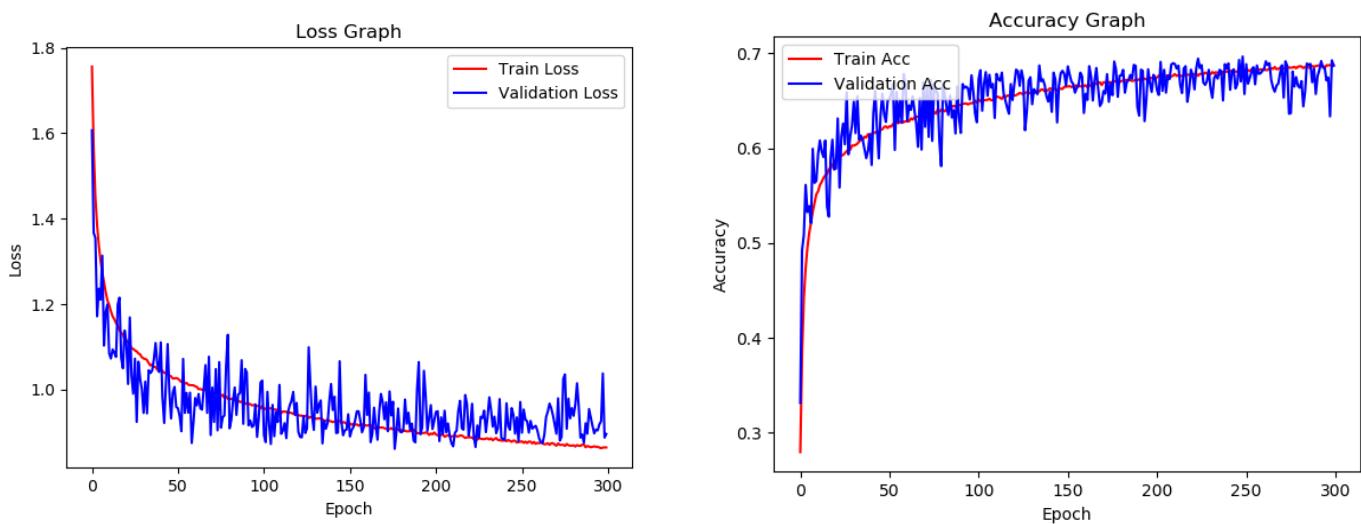
(Collaborators: 有參考去年修課的宋子維同學的做法)

在模型架構上，我共疊了四層的 Convolution Layer，分別使用了 70、140、280、560 兩倍兩倍上去的 filter 數，因第一層圖片還較大，所以使用了(5,5)的 filter，而剩下的三層 filter 大小均為(3,3)，並做 Batch Normalization 以及使用” relu” 作為 Activation function。而在每一層的 Convolution Layer 後都會加上 Maxpooling Layer，使用(2,2)的視窗大小尋找最大值，這些參數設定可以使得最後出來的圖片大小濾到剩(1,1)，方便做後續 flatten 後的預測，在 flatten 後，再加上四層的 Dense Layers，unit 數分別為最後一層 filter 數量的 560 以及 480、60、7，除了最後一層使用” softmax” 外這些 layer 也都使用” relu” 作為 Activation function。而在訓練到後來發現隨著 train accuracy 上升，validation data 的 accuracy 有下降的現象，所以又再加上了 dropout layer，前四層 Convolution Layer 的 dropout 比例分別為 0.2、0.3、0.35、0.4 逐漸上升，因認為越前面的東西越少也越接近原本圖片所以設置較小的 dropout，後四層 Dense Layer 的 dropout 比例則分別為 0.5、0.3、0.3、0，因剛開始參數很多，所以先刪了一半的 neurons，再將 dropout 比例調降，而最後一層就沒有再設置 dropout 了！以下是模型架構圖示：



而在資料預處理方面，讀入資料後先把每個input進來的train data轉換成48*48的圖片，再做normalization，也就是除以255，再來shuffle整個train data避免答案一樣的圖片在附近，然後取出後5000筆資料作為validation data，另外也有將圖片左右翻轉一同作為train data，再來做data augmentation，使用的是keras內建的ImageDataGenerator，可以達到旋轉平移縮放等的功能而產生更多train data。

而以下兩圖則分別表示了train了300個epoch的過程中train loss / validation loss關係圖以及train accuracy/ validation accuracy 的關係圖：



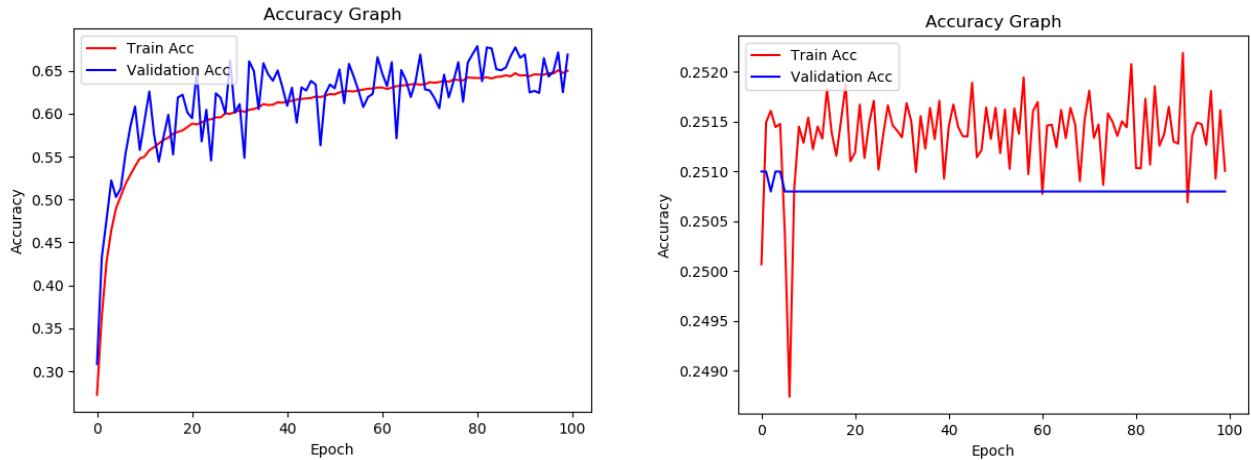
由上圖可以發現，validation data 的震盪的平均值大約跟 training data 的曲線吻合，我想這是dropout發揮了功效，避免了overfitting training data 而讓validation data的準確率下降。

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？

(1) data normalization:

我在此份 code 中實作了兩種 data normalization，第一種是一張圖內的 normalization，將整張圖片除以 255，第二種是做 batch normalization，在每層 convolution layer 後利用 keras 的套件加入 batch normalization，以下分別是有做 batch normalization/data normalization 以及沒有做 batch normalization/data normalization 在 train 了 100 個 epoch 後的準確率關係圖：

-做 Normalization/無 Normalization 的 Accuracy :

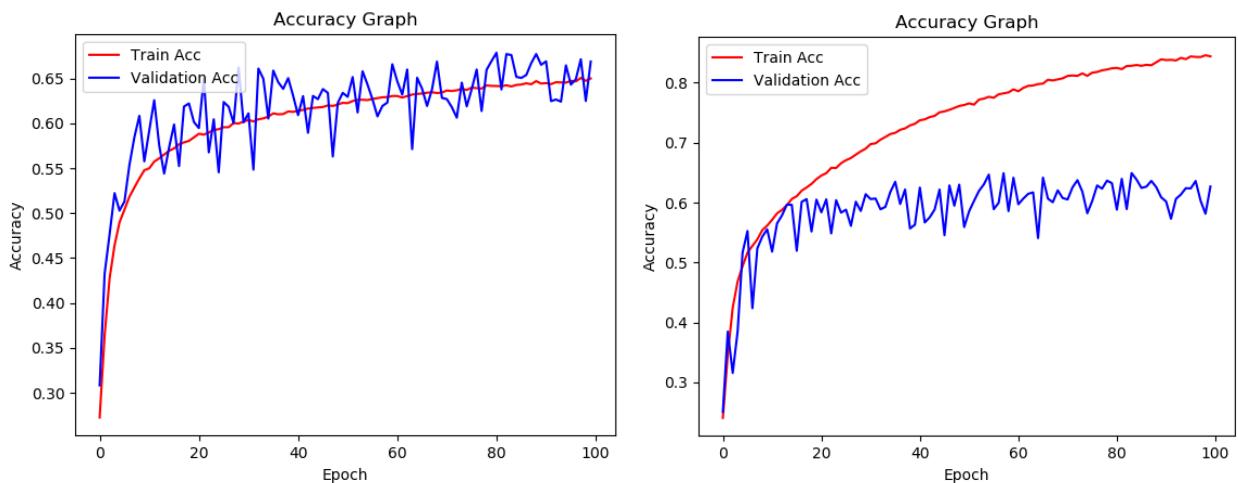


由上圖可以發現，在移除 normalization 後，training data 的 Accuracy 震盪非常大，而且沒有上升的趨勢，我想可能是因為沒有做 normalization 而導致資料異質性太大，CNN 找不到一個有跡可循的 pattern，而 validation data 甚至在幾個 epoch 之後就維持在一個低水平線，我想就是 model 沒有在 training data 得到很好的訓練所以在 validation data 準確率完全不見起色吧！

(2) data augmentation:

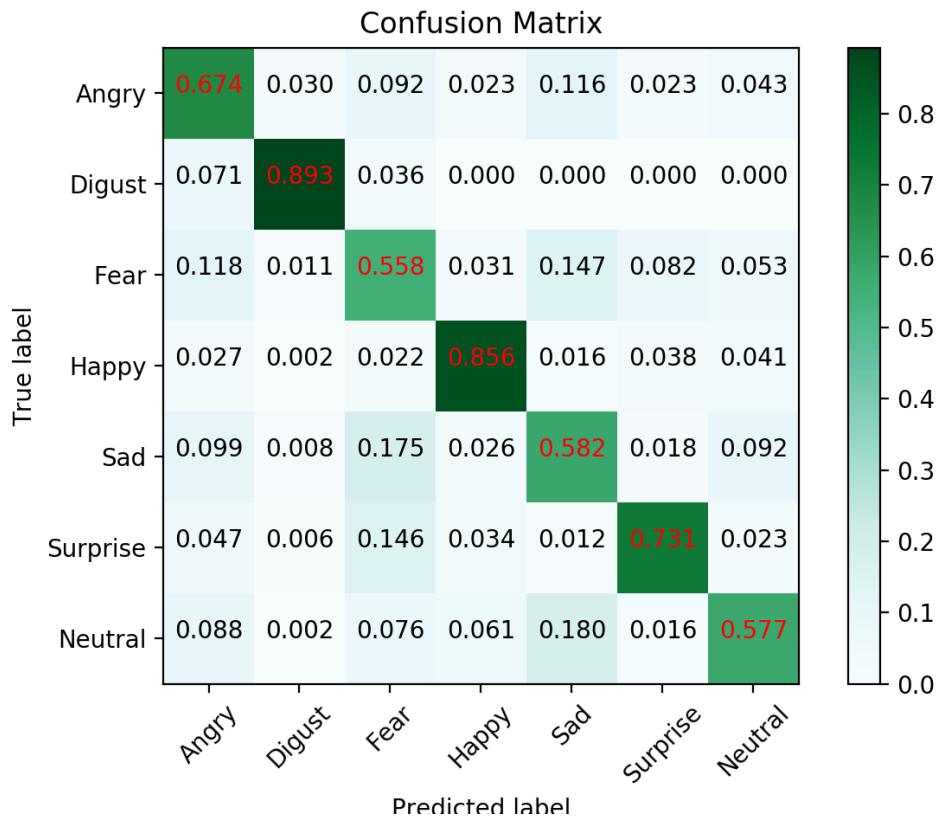
data augmentation 的實作則是利用了 keras 內建的 ImageDataGenerator，其擁有旋轉平移縮放等的功能以產生更多 train data，以下分別是有做 data augmentation 以及沒有做 data augmentation 在 train 了 100 個 epoch 後的準確率關係圖：

-做 Augmentation / 無 Augmentation :



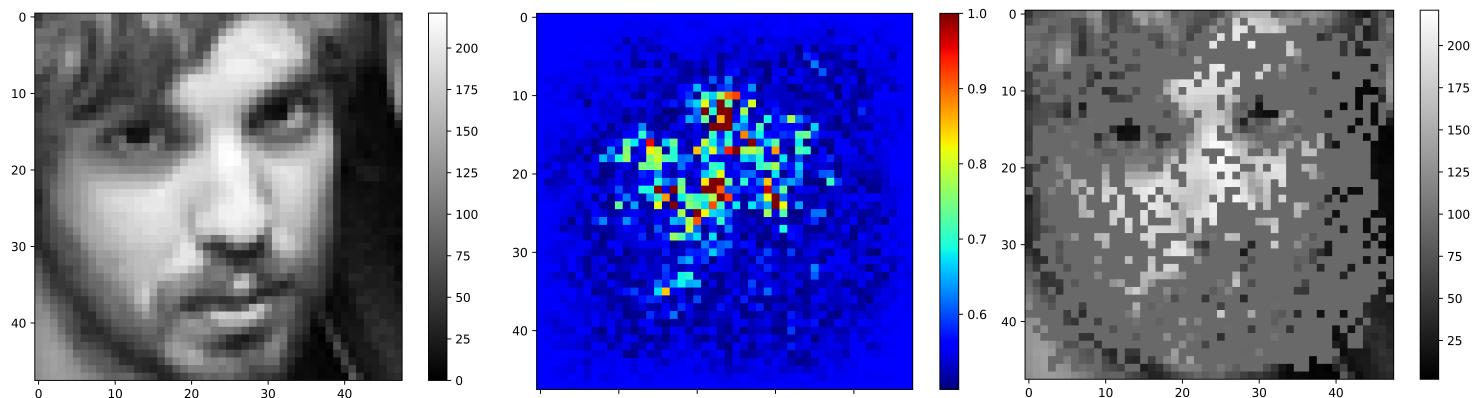
由上圖可見，在移除 data argumentation 後，雖然 training accuracy 有穩定上升的趨勢，但 validation data Accuracy 震盪的平均值則在 train 了大約十幾個 epoch 後就約略維持在一個水平線上，不見起色，我想可能是因為移除了 data argumentation，使得模型的抗震能力非常低落，對於人臉位在不同位置、不同角度的 data 就無法抓出 pattern 正確判斷。

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]

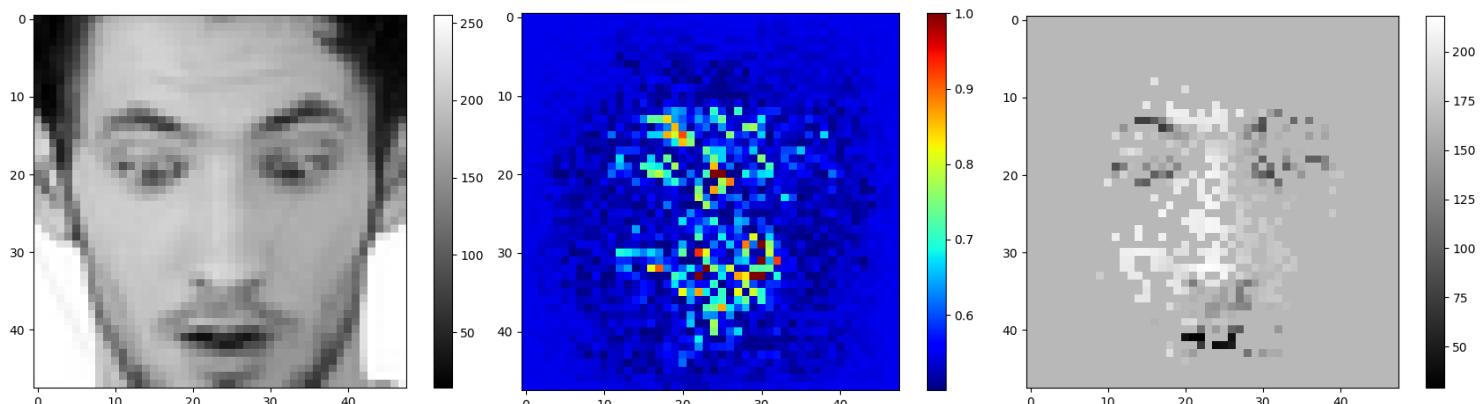


由上圖可以發現，我的 model 在 “Sad” 和 “Fear” 間很容易搞混，正確答案為 Sad 的圖片中我的 model 有 17.5% 的機率會預測為 Fear，而在正確答案為 Fear 的圖片中有 14.7% 的機率會預測為 Sad，這也可能是導致我的 model 在預測 Fear 和 Sad 的準確率偏低的緣故，另外也發現我的 model 也很容易將 Sad 判斷為 Angry，有 11.6% 的機率，以及將 Angry 判斷為 Fear，這也有 11.8% 的機率，推論可能是因為聚焦在嘴巴多半是呈現倒 U 型而有這樣的結果，又或是我們一般在判斷一個人是害怕還是傷心時本來也都會有誤判的狀況，它很容易搞混。

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？



上圖為一中立表情的圖片，此為預測正確的圖片，可以發現 model 幾乎只聚焦在眼睛鼻子的部分和外圍臉的輪廓，猜測可能是因為中立的表情嘴巴部位沒有什麼參考價值，我自己判斷也是會先從眼睛判斷！

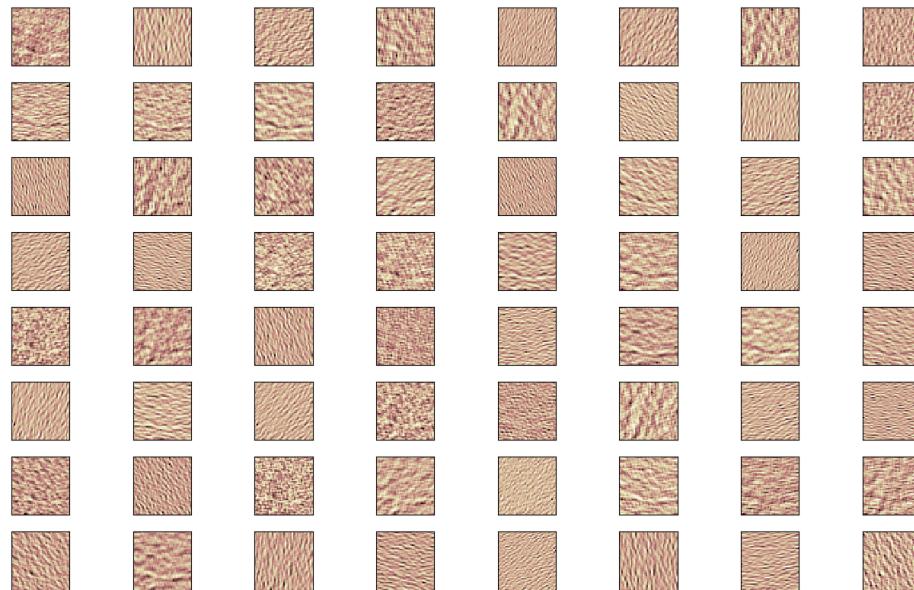


上圖為一驚訝表情的圖片，我的 model 也有正確預測出來，可以發現 model 聚焦的部分在眼睛、鼻子嘴巴，可以假定其為 model 判斷的標準，我自己看也是會多注意嘴巴這個部位而判斷為驚訝的。

綜合以上，發現 model 在判斷一張人臉時，會聚焦在人臉的輪廓、眼睛、鼻子和嘴巴以當作判斷標準，和人類在判斷一張人臉照片時的情況還滿像的！

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。

Filter :

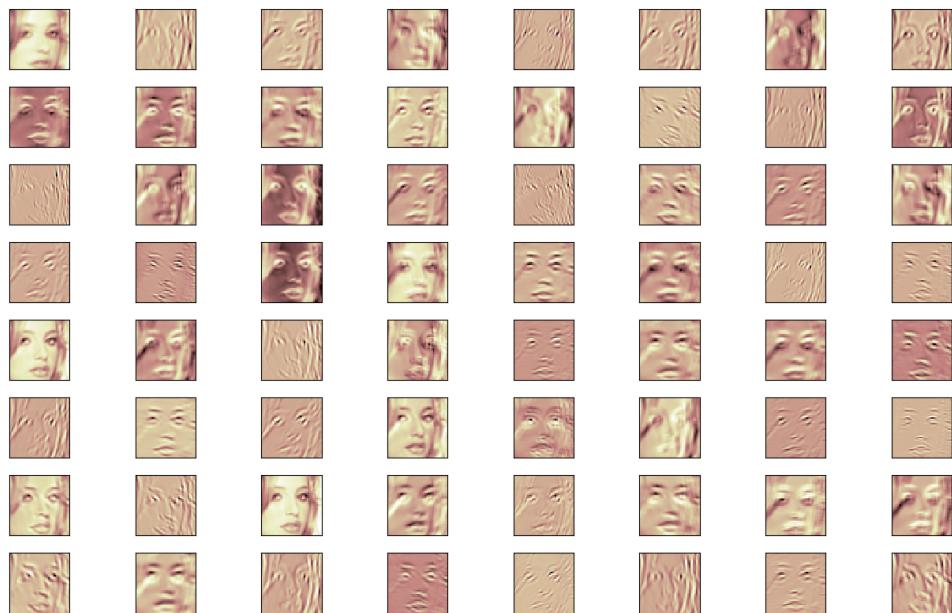


上圖為第一層 Convolution layer ($\text{Conv2D}(70, (5,5))$)取前 64 個 filter 的結果，可以很清楚地發現其多為較粗的紋路，也都以線條為主沒有看到什麼圓圈，我猜是因為他是第一層的關係，還沒有透過資料訓練到什麼所以先抓取大致的紋路，以下是從 test data set 中抓取一張照片經過此 filter 後的結果。

input :



output :



可以發現出臉大致的輪廓已經被以不少 filter 抓出來了，我想應該是要像此照片一樣五官輪廓清晰且臉沒有什麼被遮擋的圖片最能 activate 此層 layer，另外因為這張照片是側臉的關係，鼻子的輪廓特別清晰所以能被 filter 很快抓出來，像我試了另外一張正面照片就沒有這樣的效果了！所以我想應該是擁有清晰五官的圖片最能 activate 此層 layer。