

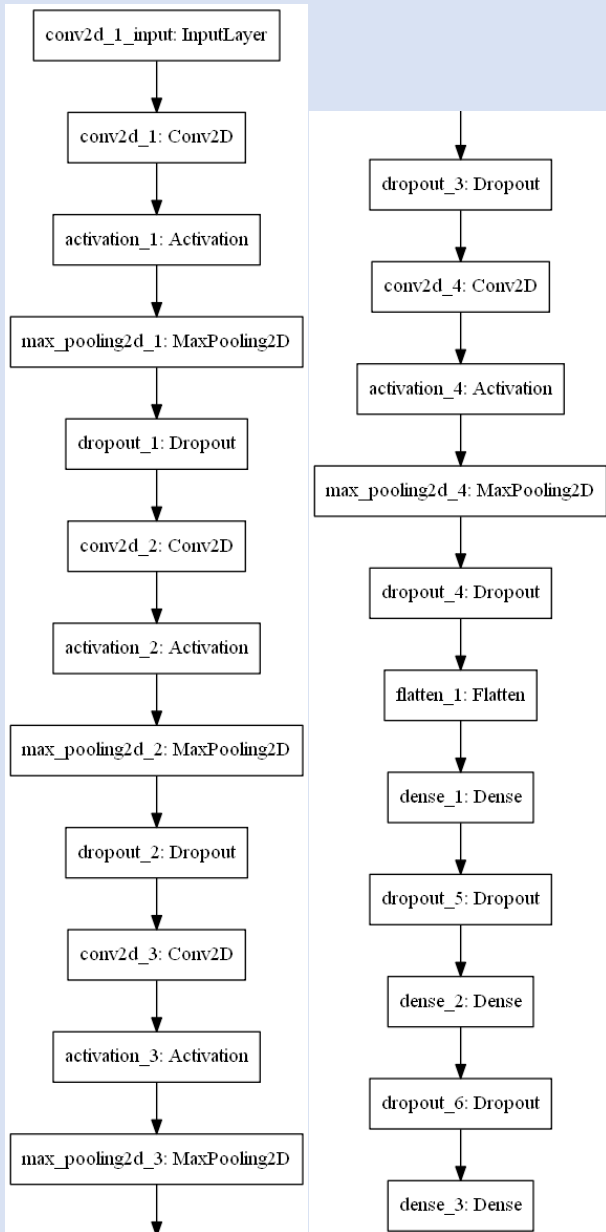
# Machine Learning HW3

學號：R04922169 系級：資工所碩二 姓名：楊智偉

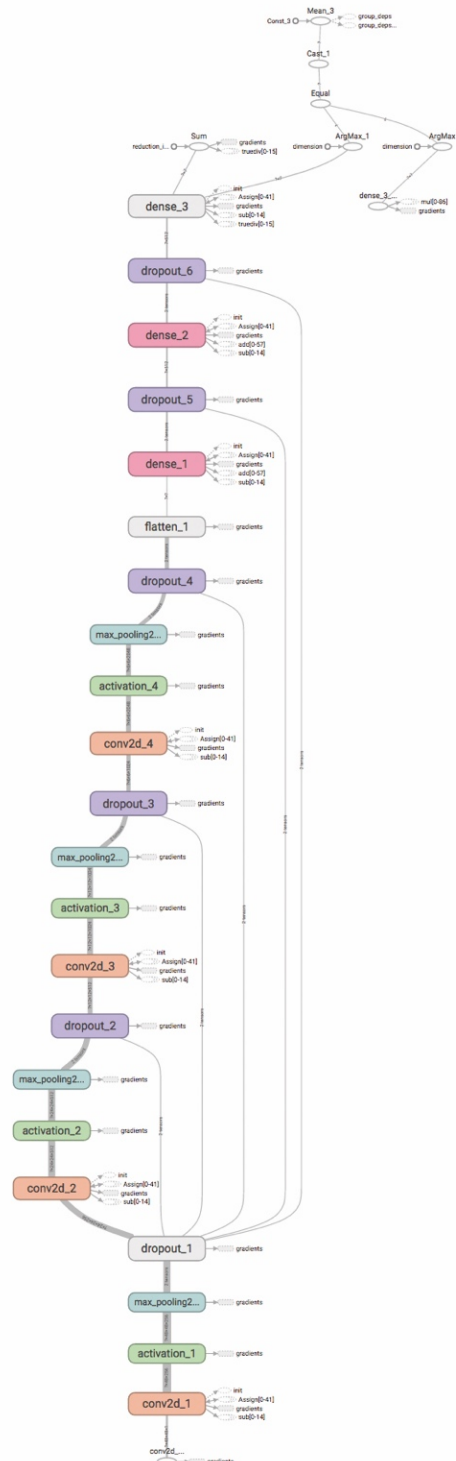
1. (1%) 請說明你實作的 CNN model，其模型架構、訓練過程和準確率為何？

答：

Keras vis\_utils



Tensorboard Graph



模型的架構方面，我首先使用四層的 Convolution 2D，filter 的數量分別是 [256, 512, 1024, 2048]。與其搭配使用的 Activation 是 relu，並且每層也會通過一個 Max Pooling 以及一個參數為 0.1 的 Dropout。

完成以上步驟後，先將資料做 flatten()，然後再通過兩個 512 維度的 Dense，來製造兩層的 fully connected neural network。程式碼的部份如下圖所示：

```
model = Sequential()
model.add(Conv2D(256, kernel_size=(3, 3), padding='same', input_shape=input_shape))
model.add(keras.layers.Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.1))

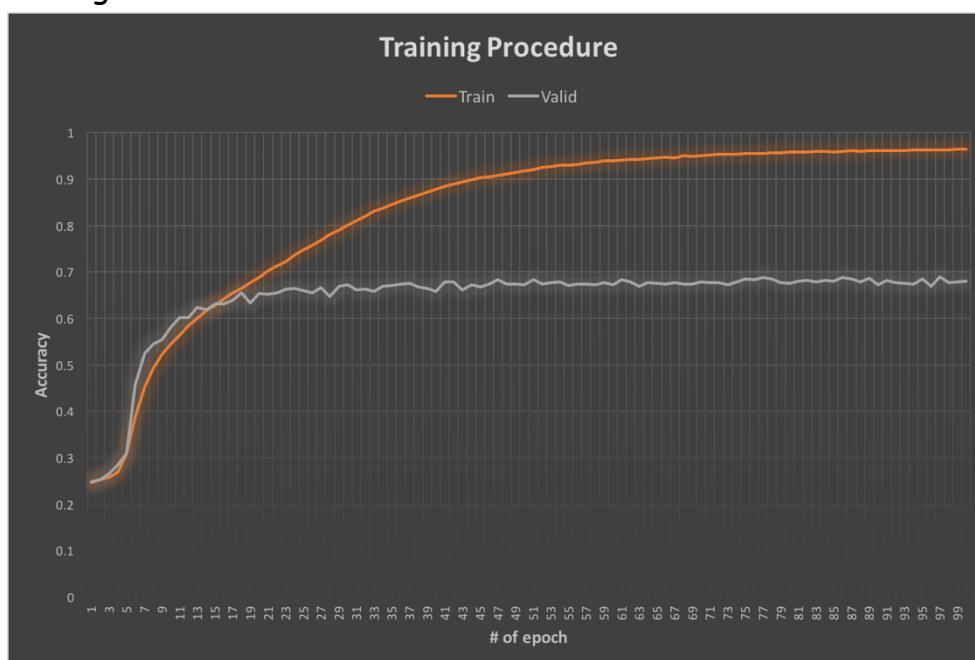
model.add(Conv2D(512, kernel_size=(3, 3), padding='same'))
model.add(keras.layers.Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.1))

model.add(Conv2D(1024, kernel_size=(3, 3), padding='same'))
model.add(keras.layers.Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.1))

model.add(Conv2D(2048, kernel_size=(3, 3), padding='same'))
model.add(keras.layers.Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.1))

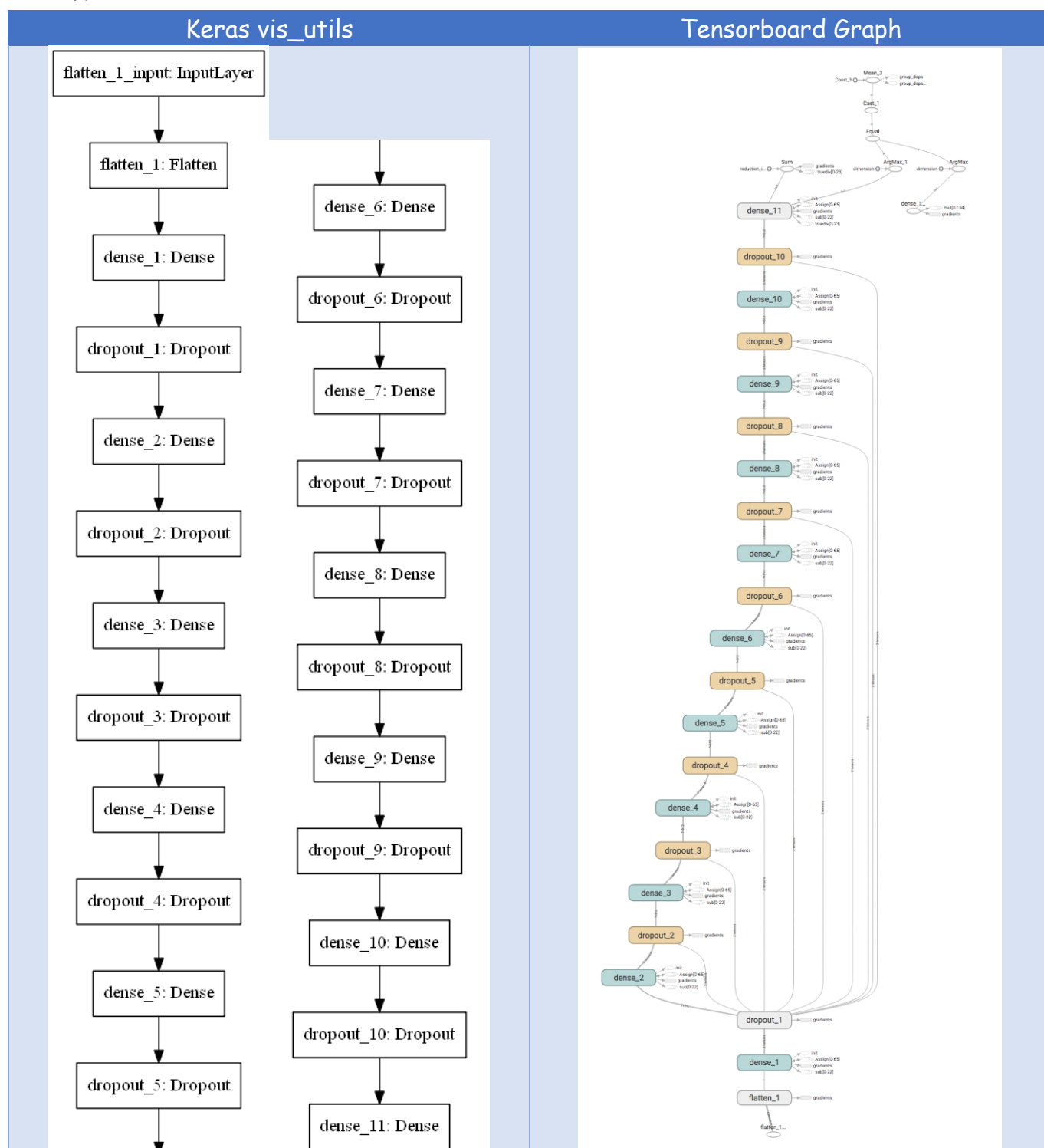
model.add(Flatten())
model.add(Dense(512, activation='relu', kernel_regularizer=keras.regularizers.l2(0.01)))
model.add(Dropout(0.5))
model.add(Dense(512, activation='relu', kernel_regularizer=keras.regularizers.l2(0.01)))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))
```

然而一開始我所測得的準確率不高，後來想到可以利用增加 training data 的方式增進 model。所以我透過 ImageDataGenerator 來製造各種位移、水平翻轉的額外 data 進行 training。Training accuracy 的結果以下圖表示，計算方式為從 training data 中切出 15% 作為 validation data。

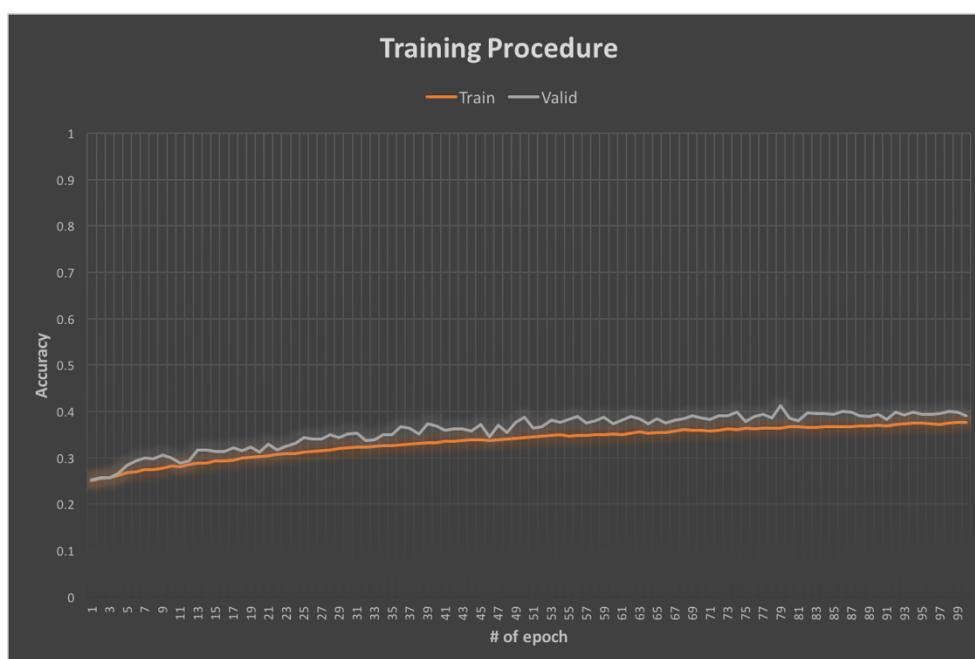


2. (1%) 承上題，請用與上述 CNN 接近的參數量，實做簡單的 DNN model。其模型架構、訓練過程和準確率為何？試與上題結果做比較，並說明你觀察到了什麼？

答：



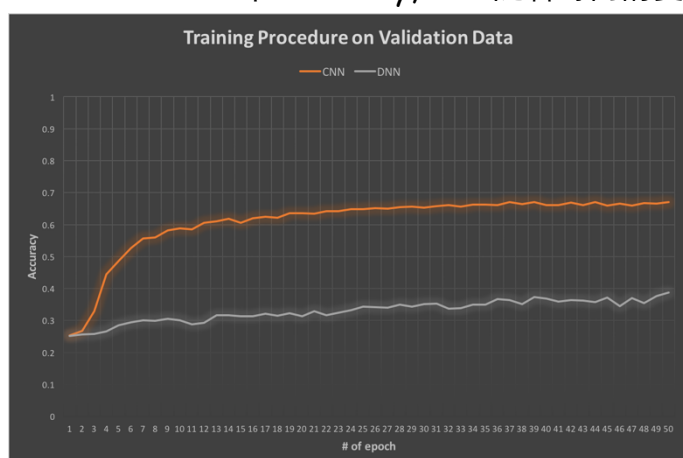
DNN model 的部分我使用 10 層 512 維度的 Dense 建造而成。每一層的 Dense 使用的 activation 為 relu，且每一層後都增加一個 0.1 的 Dropout。此 DNN model 所得到的結果如下，training set 以及 validation set 的準確度都成長的很緩慢，且幾乎貼合沒有 over fitting 的現象。



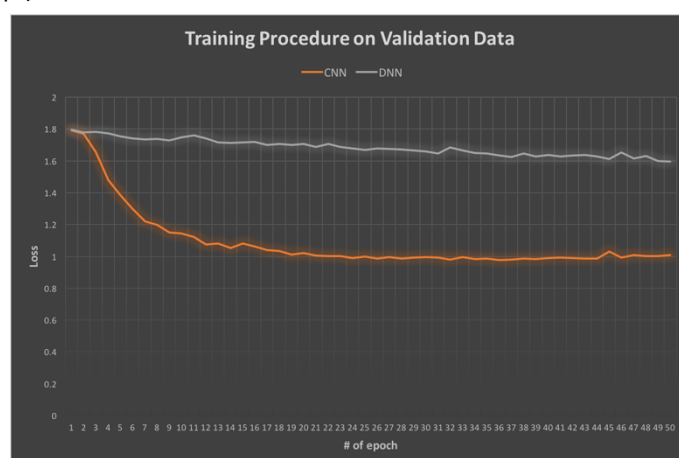
因為考慮到使用第一題 CNN model 相同量的參數來設計 DNN model 有點太多，所以我另外建立一個參數與此 DNN model 相當的 CNN model 來做比較，新的 CNN model 是減少原本 CNN model 的 filter 數量調整而來。參數的使用量以及執行結果條列於下表。

	DNN model	CNN model
<b>Total params</b>	3,547,655	3,544,519
<b>Validation Accuracy</b>	0.3875	0.6698
<b>Public Score</b>	0.3873	0.6578

接著我將 training 過程中的 50 epochs 所得到的結果，畫出在 validation data 中 accuracy, loss 隨著時間的變化圖。



Accuracy on Validation data



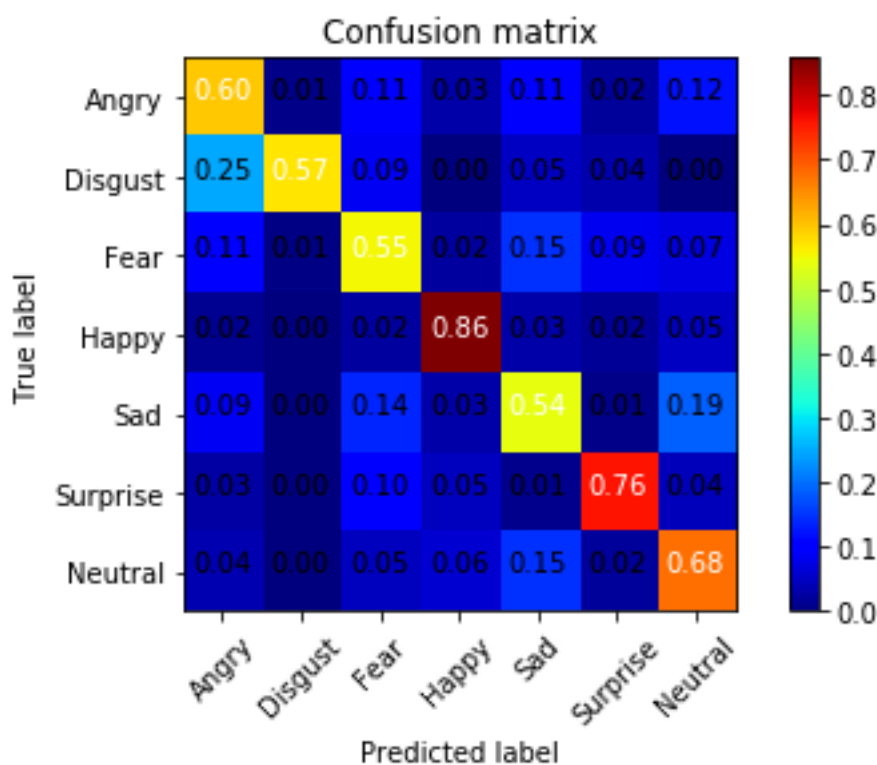
Loss on Validation data

我們可以觀察到，在相同數量的參數底下，CNN model 的表現遠比 DNN model 還要好非常多。且兩個 model 也都是在大約 30 個 epochs 後趨近平緩。而用 loss 的角度來看，CNN model 也在是在前 15 個 epochs 就能將 loss 急遽的壓得非常低，結果相當不錯。

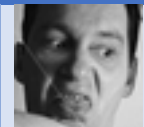
### 3. (1%) 觀察答錯的圖片中，哪些 class 彼此間容易用混？[繪出 confusion matrix 分析]


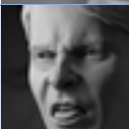
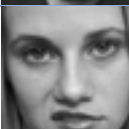
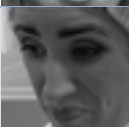
答：

我將 training data 的 15% 抽出當成 validation data，並且利用這些 validation data 來分析 confusion matrix。由以下圖片顯示可以發現，在 Disgust、Fear 和 Sad 類別的圖片最容易被 predict 錯誤。



為了分析上述容易 predict 錯誤的情況，我們隨機挑出幾張 validation set 中屬於 Disgust 類別的資料做 probability distribution 的分析，結果如下表。可以發現很多資料以人為判斷也不見得會被分到 Disgust 的類別，反而是比較接近 Angry 的類別。還有些部分的資料則是有點模凌兩可，無法確切的分好類別，所以這是在 labeled data 就已經有的情況，並且反映在 confusion matrix 中。

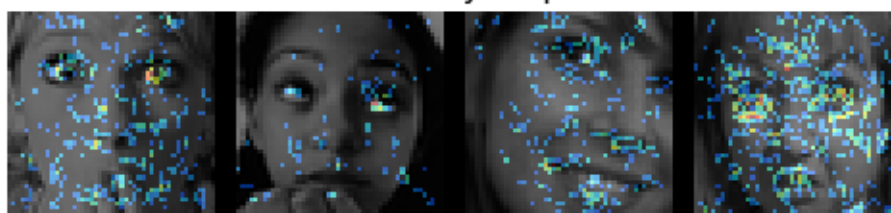
	Angry	Disgust	Fear	Happy	Sad	Surprise	Neutral
	0.95	0.02	0.01	0	0.02	0	0

	0.22	0.08	0.46	0.01	0.22	0.01	0
	0.99	0	0.01	0	0	0	0
	0.01	0.97	0	0	0.02	0	0s
	0.01	0	0.01	0	0.98	0	0

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：

Saliency map



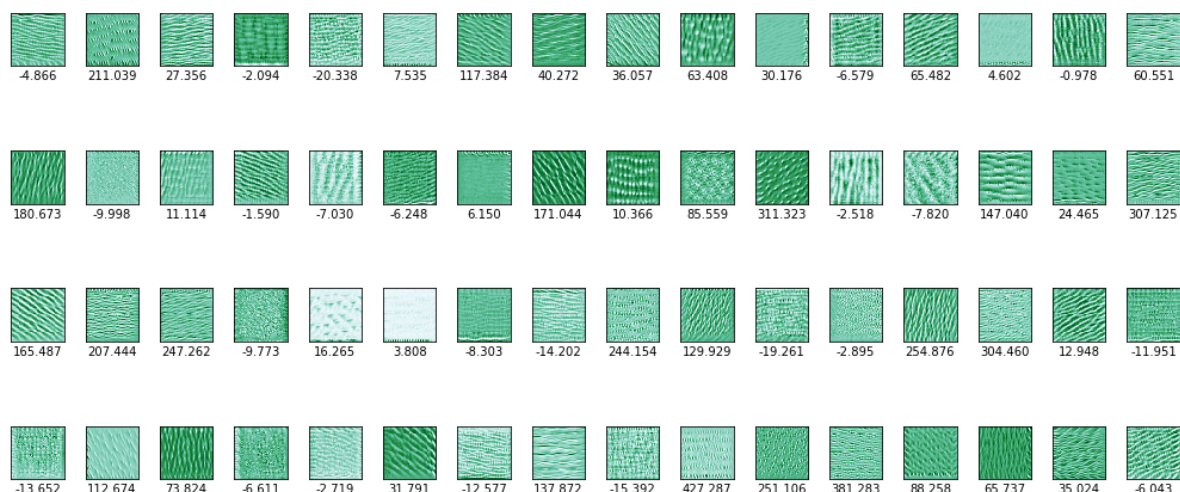
我自 CNN model 中挑出一層 conv2d layer 來進行觀察。上圖是隨機選出四張資料來計算 Saliency map 的結果。在多數資料中可以觀察到『眼部』以及『嘴部』感覺是最容易被 focus 的部分，顯而易見地因為人類的表情大多可以由眼神以及嘴部肌肉的牽扯來分辨。除此之外，眼睛周圍以及臉頰的部分也常常會有被 focus 的情況發生。

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。

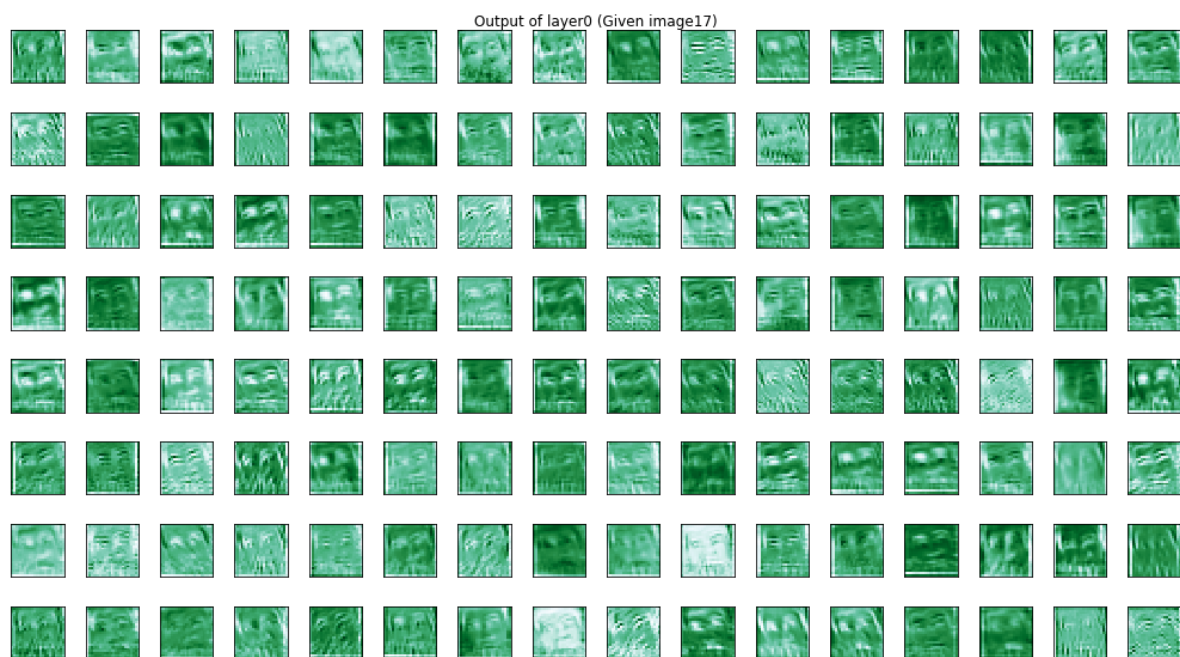
答：

我利用 gradient ascent 的方式，找出什麼樣的 image 最容易能夠 activate 特定層的 filter。目標是 CNN model 中的第二個 conv2d layer，結果如下。結果發現大多數的 image 都是比較偏向各種方向的條紋圖片，反而很少出現如助教解說中有幾張有模糊的臉型的結果。推測是因為我所採用的 filter 是屬於比較淺的第二層，activate 出來的東西是比較偏向於簡單的 pattern，而在較為深層的 filter 可能就會有較特別的結果。





另外，我挑了一張  來測試經過那些 **filter**，我們所得到的影像會呈現什麼樣的結果，此 **image** 是屬於 **Fear** 的分類中。通過 **filter** 之後的 **output** 結果如下。大多數的結果可以看出臉部的形狀，也有很大一部份能夠把表情中的重要特徵擷取出來。



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

[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%]

- you can use other model with poor performance to see what is the difference

我去找了另外一個相對準確度較低的 CNN model 來做比較，他在 validation data 上的準確率是 0.55，與第五題的 0.68 準確率有些差距。

首先，第一個差別是 filter 的數量，高準確率的 filter 數量使用較多，參數也使用較多。另外，可以觀察到此 model 所能保留的表情特徵稍少，並不太容易使用這些結果來推測此 image 依該是屬於哪些分類的。

