機器學習作業三 report

學號：B05901040　系級：電機三　姓名：蔡松達

1. (2%) 請說明你實作的 CNN model，其模型架構、訓練參數和準確率為何？並請用與上述 CNN 接近的參數量，實做簡單的 DNN model，同時也說明其模型架構、訓練參數和準確率為何？並說明你觀察到了什麼？
   (Collaborators: 無)

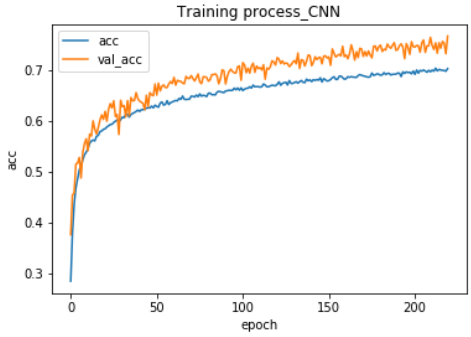
答：模型架構如下表所示，兩邊都是跑 220 次 epochs，batch size 皆設為 100，在 CNN 上有實作 augmentation，其做法與第三題所做的相同，生成的資料量與原先的 training data 量相同，原先的 training data 則做為 validation set；DNN 上則並沒有使用，僅使用 10%的 training data 作為 validation set，兩邊的參數也接近相同，而下面也列出兩者在 test data 上的 accuracy，從第二題的圖中可以發現 CNN 的 training data 雖然 accuracy 成長的較慢，但是 validation 的 accuracy 卻始終高於 training data 的 accuracy，兩者都呈現成長的趨勢；而 DNN 雖然在 training data 上的 accuracy 成長得較快，但 validation 的 accuracy 卻完全沒有任何進步，這說明基本上使用簡單的 DNN 架構並沒有辦法讓 model 真正 train 起來，而只是 overfit 在 training data 上，而較深層的 CNN 則比較能夠真正 train 起來，這樣的情形也可以從 testing accuracy 中看的出來；此外 DNN 的架構 train 的速度比起 CNN 快了不少也是可以相當容易觀察出來的現象。（loss: categorical crossentropy）

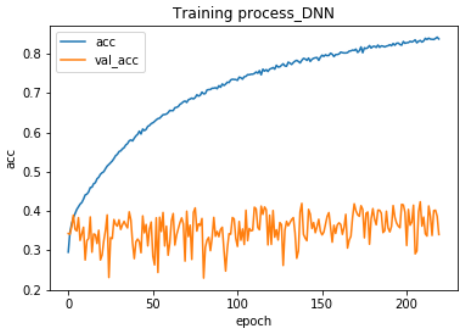
Model	CNN model	DNN model
Model 架構	總參數量：1332807 在此架構中每三層相同的 Convolution 後面會接一層 Batch normalization 層（參數採預設值），後面再接一層 LeakyRelu(alpha=0.3) 作為 activation，接著使用一層 Maxpooling，最後再接一層 Dropout(rate=0.3)，總共搭建三組如此的架構，以下僅寫出三組中 Convolution 及 Maxpooling 層各自的參數： 第一組：Conv2D(32, kernel_size = (3, 3), strides=(1, 1), padding='same' , input_shape = (48, 48, 1)) MaxPooling2D(pool_size=(2, 2), strides = (2,2), padding='same') 第二組： Conv2D(64, kernel_size = (3, 3), strides=(1, 1), padding='same') MaxPooling2D(pool_size=(2, 2), strides = (2,	總參數量：1316359 Dense(512, input_dim=2304)) BatchNormalization() Dropout(0.3) LeakyReLU(alpha=0.3) Dense(units = 256) BatchNormalization() Dropout(0.3) LeakyReLU(alpha=0.3) Dense(units = 7, activation = 'softmax')

	2), padding='same') 第三組：Conv2D(128, kernel_size = (2, 2), strides=(1, 1), padding='valid') MaxPooling2D(pool_size=(2, 2), strides = (2, 2), padding='valid') 最後接上下列幾層： Flatten() Dense(units = 512) BatchNormalization() Dropout(0.3) LeakyReLU(alpha=0.3) Dense(units = 7, activation = 'softmax')	
Public score	0.69100	0.39732
Private score	0.67316	0.39788
Testing score	0.68208	0.39760

2. (1%)承上題，請分別畫出這兩個 model 的訓練過程 (i.e., loss/accuracy v.s. epoch) (Collaborators: 無)

答：此處比較 accuracy v.s. epoch，而 loss 的部分是使用 categorical crossentropy，基本上 training data 的 loss 隨著 epochs 次數都會越來越低，但只有 CNN 的 validation loss 也是下降的趨勢；DNN 的 validation loss 基本上完全呈現震盪的狀態，代表根本沒 train 起來；accuracy 呈現的趨勢也與 loss 的狀況相符合。





3. (1%) 請嘗試 data normalization, data augmentation,說明實作方法並且說明實行前後對準確率有什麼樣的影響？(Collaborators: 無)

答：data normalization 是將所有數值除以 255 使值落在 0~1 之間，而此處 data augmentation 則是使用隨機的旋轉（範圍從 0 到 20 度）、隨機的左右平移（最多平移寬度的 10%)、隨機的上下平移(最多平移高度的 10%)、隨機剪切(shear)、隨機放大(zoom)以及隨機的水平翻轉，準確率如下表所示（其餘架構與第一題的 CNN 除了 epochs 改為 50 次外，皆完全相同）：

方式	Public score	Private score	Average score
未使用兩種方式	0.64419	0.62719	0.63569
僅使用 normalize	0.64391	0.63193	0.63792
兩種方式皆使用	0.61382	0.61660	0.61521

在做這個部分的實驗時可以觀察的出來其實有沒有使用 normalize 對於最終結果影響並不大，可以從前兩個 row 的結果看的出來，並沒有太大的區別，可能的原因是因為我在 CNN 架構中本身就已經擺放了 Batch normalization 層，針對每一層 layer 的 output 都進行標準化，因此起初的 normalize 能彰顯的效果就變的不明顯；但是若是使用了 augmentation 的方法後卻讓結果有了明顯的轉變，在此處看到的結果變差了一些，主要是因為我在此處只設了 50 個 epoch，使用 data augmentation 通常需要 train 比較多的 epoch 才會得到較好的結果，此外也可能是因為這裡設定的 augmentation 參數值不見得是非常好的，可能需要嘗試更多種類的參數來增強效果；而 training 過程中其實可以發覺有使用 data augmentation 的時候比較不容易 overfit，因為圖形經過不同的轉換，可以獲知較多的資訊，因此若是嘗試更多的 epoch 數想必會有不錯的結果（從第一題的結果可以觀察的出來），可知 augmentation 對於結果影響甚鉅。

4. (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](Collaborators: 無)

答：最容易判斷錯誤的例如恐懼判斷成難過；難過或生氣判斷成中立，但其實中立自己本身答對的狀況並不差，只是有的時候難過跟生氣的狀態可能會沒那麼容易判定準確因而被判定為中立；恐懼本身判斷的能力是最差的，也因此恐懼也是相當容易與其他混淆的，而恰好生氣與難過也是自身判斷能力較差的，可見容易被混淆的 label 可能代表該 label 較難被判斷出來（生氣、恐懼、難過），或是該 label 是接收任何不明顯的其餘圖片（中立）。