

機器學習作業三 報告

學號 - B03902015 / 系級 - 資工三 / 姓名 - 簡瑋德

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

模型架構

- Input Layer - shape = (1, 48, 48)
- Convolution2D - n_filters = 64, kernel_size = (3, 3)
- Activation - 'relu'
- Convolution2D - n_filters = 64, kernel_size = (3, 3)
- Activation - 'relu'
- MaxPooling2D - pool_size = (2, 2)
- Dropout - rate = 0.25
- Convolution2D - n_filters = 64, kernel_size = (3, 3)
- Activation - 'relu'
- MaxPooling2D - pool_size = (2, 2)
- Dropout - rate = 0.25
- Flatten
- Dense - units = 512
- Activation - 'relu'
- Dropout - rate = 0.48
- Dense - units = 256
- Activation - 'sigmoid'
- Dropout - rate = 0.48
- Dense - units = 7
- Activation - 'softmax'

參數量

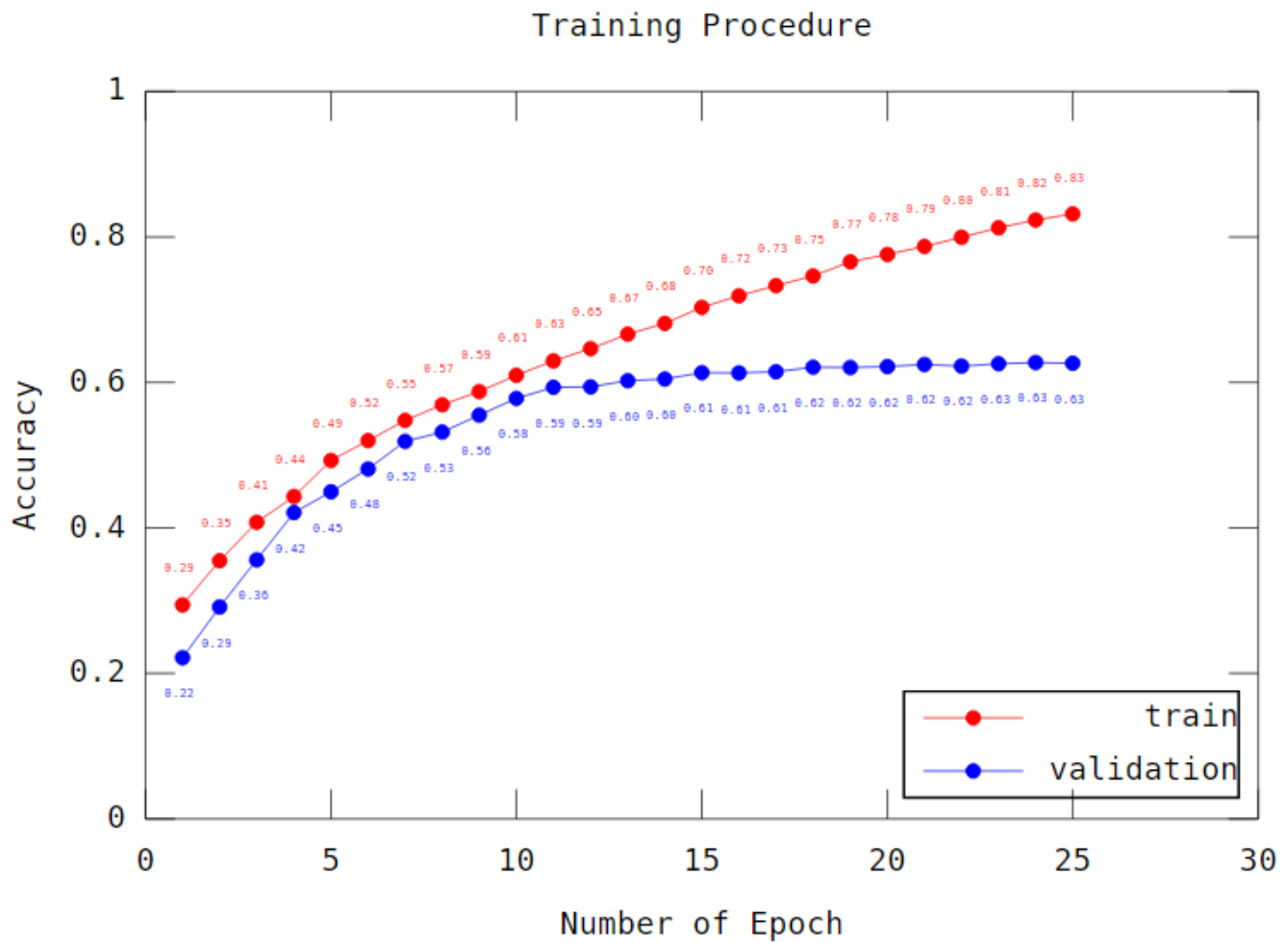
$$(1 \times 64 \times 3 \times 3 + 64) + (64 \times 64 \times 3 \times 3 + 64) + (64 \times 64 \times 3 \times 3 + 64) \\ + (10 \times 10 \times 64 \times 512 + 512) + (512 \times 256 + 256) + (256 \times 7 + 7) \approx 3,480,000$$

訓練過程

- 損失函數以 'categorical_crossentropy' 計算
- 優化器使用 'adadelata'
- Epoch數設為25

- Batch大小設為128
- 會把訓練用的圖片左右翻轉，以增加訓練圖片的數量

準確率變化圖



2. 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？

模型架構

- Input Layer - shape = (2304)
- Dense - units = 1024
- Activation - 'relu'
- Dropout - rate = 0.25
- Dense - units = 512
- Activation - 'relu'
- Dropout - rate = 0.25
- Dense - units = 512
- Activation - 'relu'
- Dropout - rate = 0.25

- Dense - units = 7
- Activation - 'softmax'

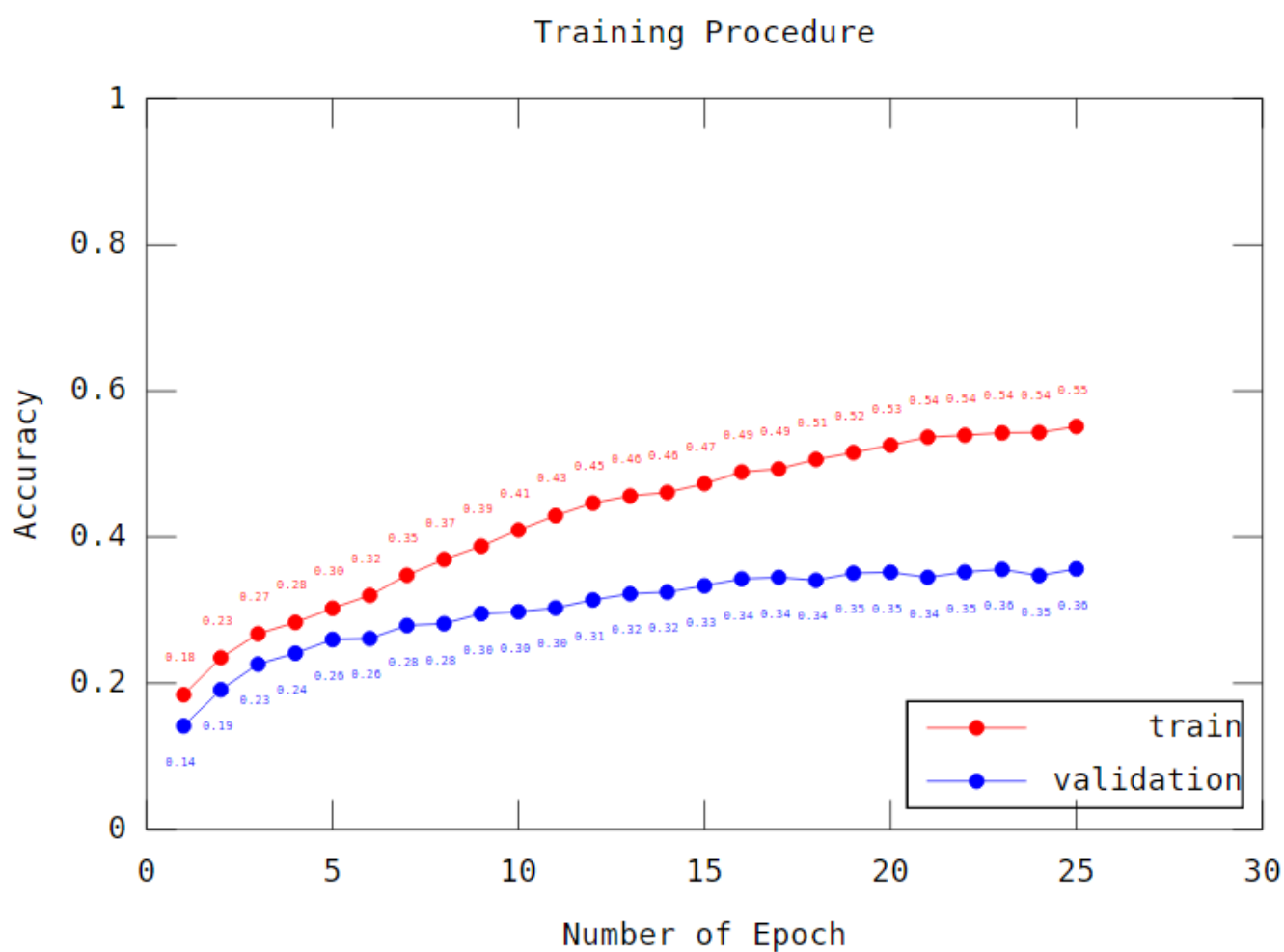
參數量

$$(2304 \times 1024 + 1024) + (1024 \times 512 + 512) + (512 \times 512 + 512) + (512 \times 7 + 7) \\ \approx 3100000$$

訓練過程

- 損失函數以 'categorical_crossentropy' 計算
- 優化器使用 'adadelat'
- Epoch數設為25
- Batch大小設為128
- 會把訓練用的圖片左右翻轉，以增加訓練圖片的數量

準確率變化



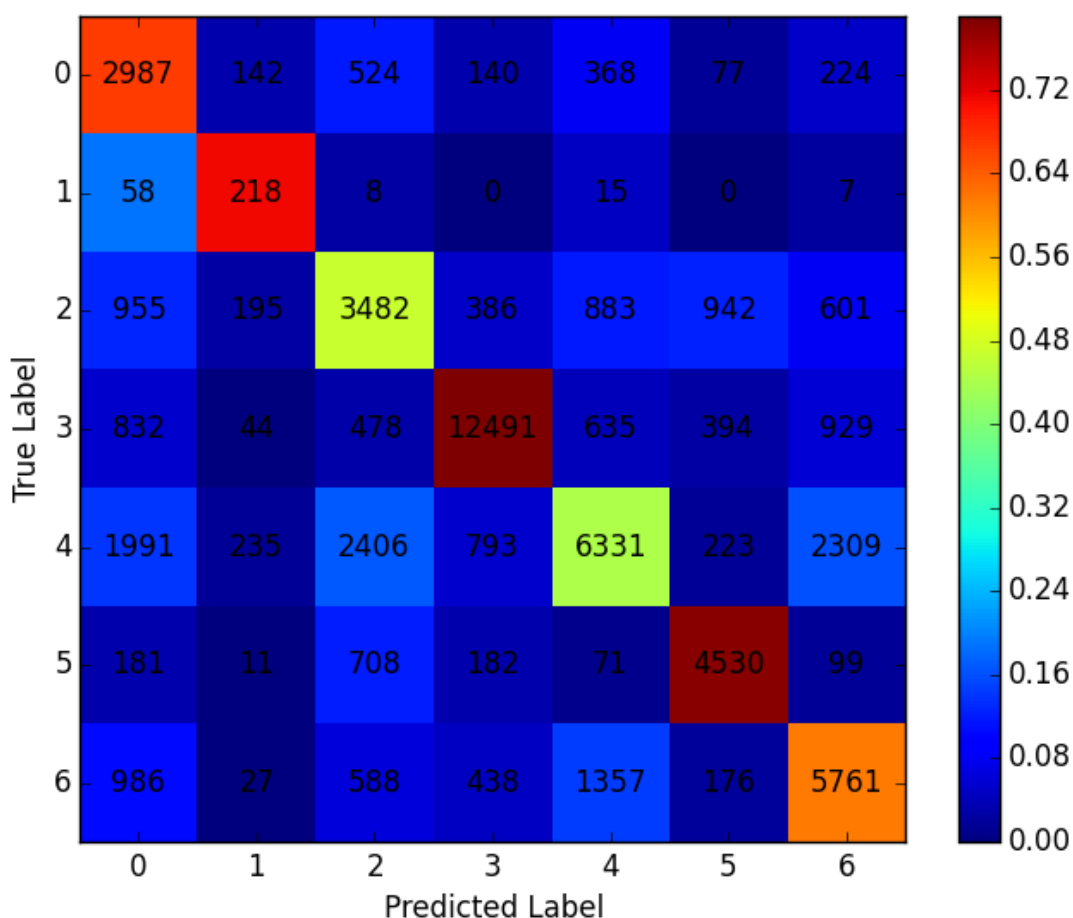
與「CNN」的比較

- 參數量接近時，「CNN」的效果會比「DNN」好

- 就結構上來說，「CNN」多了「Convolution」和「Pooling」這兩個操作，對影像來說，更能有效抓出圖片的特徵
- 「Convolution」和「Pooling」著重於相鄰像素之間的關係，而不是把圖片壓平對每個像素一視同仁，此外，也有降低維度的效果，把省下來的參數用在其他的地方，以增加訓練的準確度

3. 觀察答錯的圖片中，哪些「class」彼此間容易用混？(繪出「confusion matrix」分析)


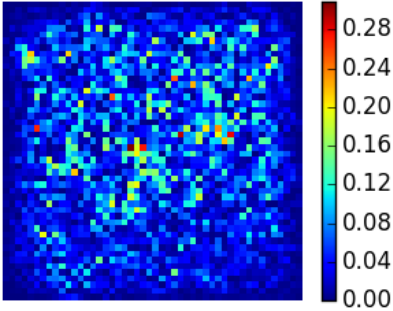

Confusion Matrix



分析

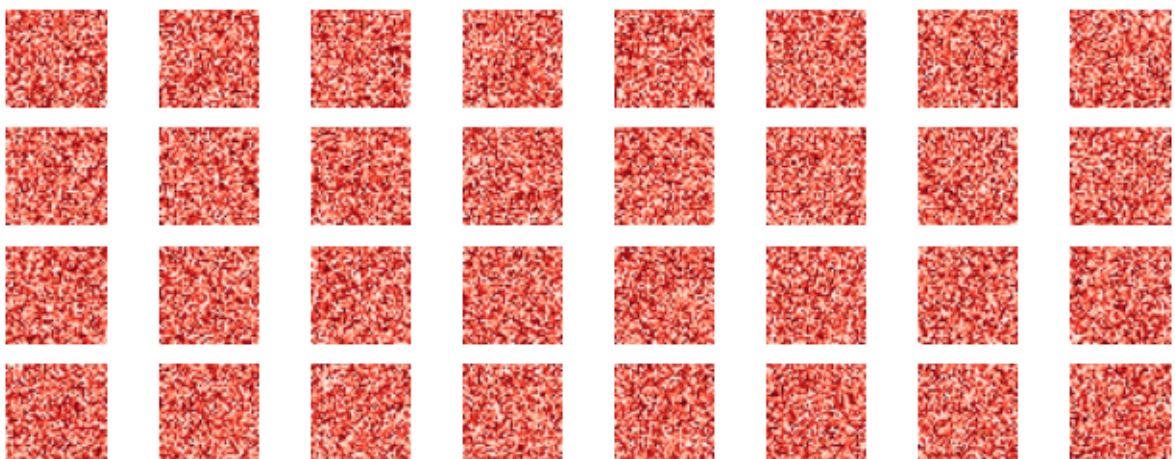
- class3的圖片比較容易被區分出來，我認為這是因為它的樣本數較多，學習的機會比較多
- class2和class4是比較容易混淆的，在訓練時模型就沒辦法區分得很清楚，答對的機率甚至只有五成左右
- 從人的角度來想，class2和class4分別是恐懼和難過，這兩個情緒表現的相似度原本就偏高

4. 從1、2題可以發現，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？

Origin Image	Saliency Map	Masked Image
		

5. 承1、2題，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」

- 第「一」層「CNN」的32個filter，分別易被以下圖片激活



- 可以發現大部分的圖片都還是「WhiteNoise」，推測可能的原因是 - 目標函數是 $a^k = \sum_{i=1}^{46} \sum_{j=1}^{46} a_{ij}^k$ ，每個像素對目標函數的貢獻其實是相同的（都會被filter的每一格乘到一次）舉例來說，若filter是[1, 2, 3; 4, 5, 6; 7, 8, 9]，任何一個pixel增加1，目標函數都會上升 $1 + 2 + \dots + 9 = 45$

6. 從「training data」中移除部份「label」，實作「semi-supervised learning」

實作方式

1. 將資料依4 : 1 : 1切成「training/unlabel/validation data」
2. 先在「labeled training data」訓練40個epoch
3. 對「unlabel data」進行一次預測
4. 這時候，每一張未標記圖片都有以下的資訊：「各個class的機率」以及「機率最大的class」

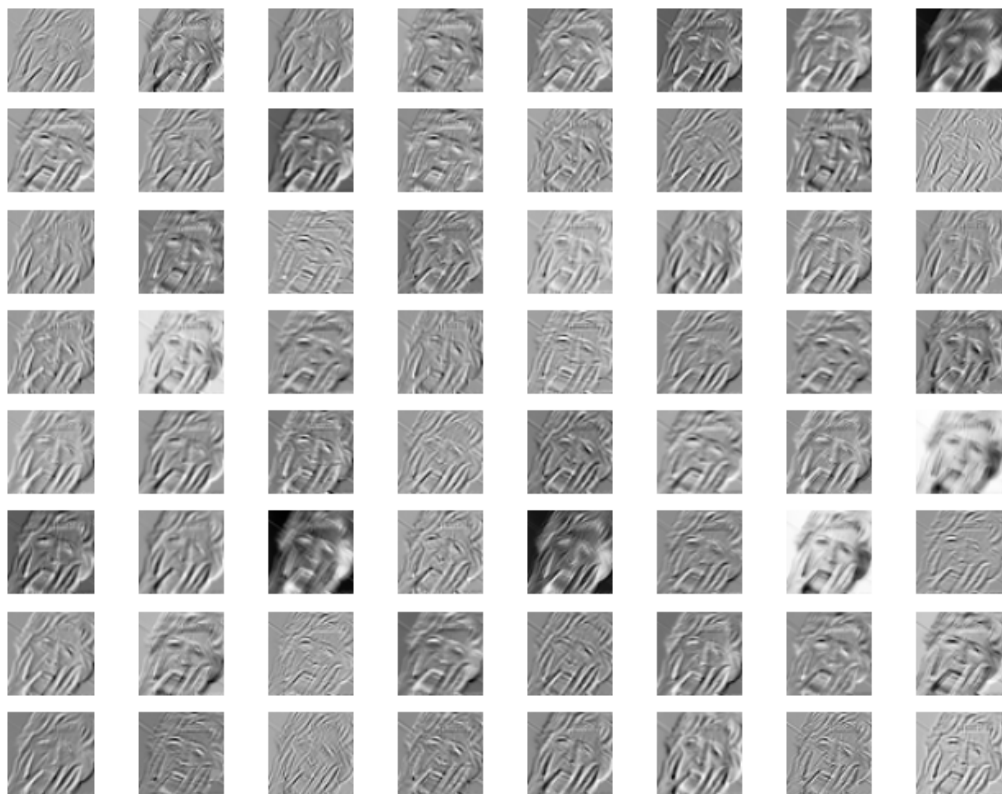
5. 若某一張未標記圖片，其「機率最大之class的機率值」大於0.9，就將它加入訓練資料集，在下一階段使用，反之則略過
6. 在「labeled data」和「pseudo labeled data」組成的訓練資料集上，再訓練10個epoch
7. 再進行一次預測，並把結果視為最終的輸出

準確率的比較

- 無semi-supervised、40個epoch - Validation Acc = 0.6443
- 無semi-supervised、50個epoch - Validation Acc = 0.6518
- 有semi-supervised、50個epoch - Validation Acc = 0.6602

7. 在「Problem 5」中，提供了3個「hint」，可以嘗試實作及觀察

- 以下是第「一」層「CNN」中，64個filter的輸出



- 觀察後可以發現，有滿多filter的結果是幾乎相同的，我認為這可能表示該層的filter數目其實不需要這麼多（並沒有新的資訊被filter抓出來）
- 此外，也可以發現，由於這是第一層「CNN」，大部分的圖形和原本的圖片仍很相近