

Homework3 Report Template

Professor Pei-Yuan Wu
EE5184 - Machine Learning

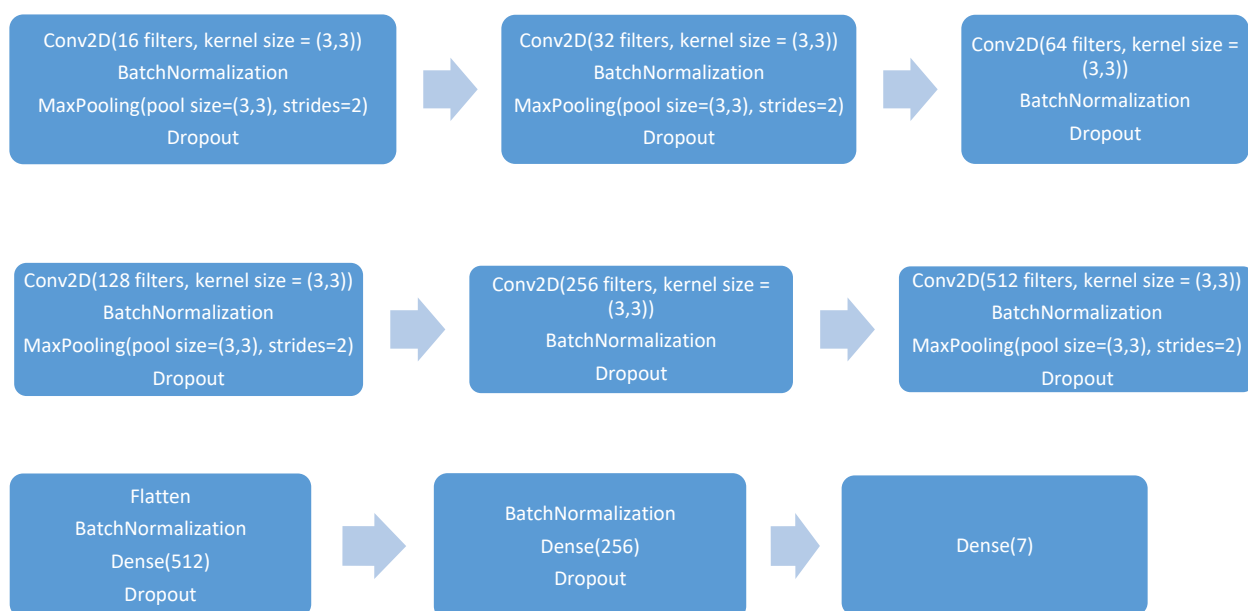
姓名：魏佑珊

學號：B05902074

Note: 1~3 題建議不要超過三頁

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？

模型架構：



主要是以 6 層 Conv 和 3 層 Dense 組成，6 層 Conv 分別有 16, 32, 64, 128, 236, 512 個 filter；Dense 則分別有 512, 256 個維度，最後利用 softmax 壓到 7 維。我用 batch size 等於 150，epoch = 300 來 train，這樣達到的準確度大約是: public 0.64363 private 0.65032。後來也有再加上 ImageDataGenerator 和 Ensemble，就能過 strong baseline 了。

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？

我以(由前到後)units=128, 256, 1024, 2048, 7 的 Dense 來架構 DNN。當然中間都有隔著 BatchNormalization Layer 和 Dropout Layer。Epoch = 300, batch size = 150 (同第一題的 CNN)

CNN 參數: 2,768,967

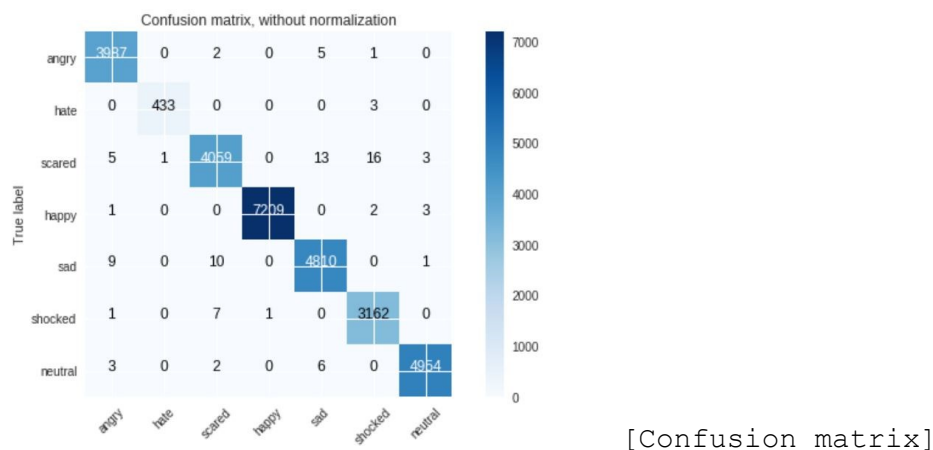
CNN 準確率 public 0.64363 private 0.65032

我建的 DNN 參數: 2,718,599

DNN 準確率 public 0.36054 private 0.36333

由上面的數據看出，DNN 很明顯的表現不如 CNN，訓練時 accuracy 也爬升的極慢。因此，訓練與 image 有關的 model 時，CNN 常常都還是較佳的選擇。

3. (1%) 觀察答錯的圖片中，哪些 class 彼此間容易用混？並說明你觀察到了什麼？[繪出 confusion matrix 分析]



[Confusion matrix]

觀察 confusion matrix 我發覺 hate 占整體資料量最少，最多的則是 happy。

至於容易搞混的類別，大約有以下幾種(<True--Predicted>) (照容易搞混的程度由大到小排序)：

<scared--shocked>, <scared--sad>, <sad--scared>, <sad--angry>, <shocked--scared>, <neutral--sad>

我發覺驚嚇和恐懼、驚嚇和悲傷最容易被搞混，其實蠻合理的，因為這幾者可能都有瞪大眼睛、張大嘴巴等等的類似面部動作；而快樂最不容易和其他的類別搞混。

-----Handwritten question-----

4. (1.5%, each 0.5%) CNN time/space complexity:

For a. b. Given a CNN model as

```

model = Sequential()
model.add(Conv2D(filters=6,
                  strides=(3, 3),
                  padding = "valid",
                  kernel_size=(2,2),
                  input_shape=(8,8,5),
                  activation='relu'))
model.add(Conv2D(filters=4,
                  strides=(2, 2),
                  padding = "valid",
                  kernel_size=(2,2),
                  activation='relu'))

```

And for the c. given the parameter as:

```

kernel size = (k,k);
channel size = c;
input shape = (n,n);
padding = p;
strides = (s,s);

```

- a. How many parameters are there in each layer(Hint: you may consider whether the number of parameter is related with)

```

Layer A:
(5 * 2 * 2 + 1)*6 = 126
Layer B:
(6 * 2 * 2 + 1)*4 = 100

```

- b. How many multiplications/additions are needed for a forward pass(each layer).

```

Layer A:
<input: 8*8*5>
multiplication:6 * (2*2*5*9)=1080
addition:6 * (2*2*5-1)*9 = 1026
Layer B:
<input: 3*3*6>
multiplication:4 * (2*2*6*1) = 96
addition:4 * (2*2*6-1)*1 = 92

```

- c. What is the time complexity of convolutional neural networks?(note: you must use big-O upper bound, and there are l (lower case of L) layer, you can use C_l, C_{l-1} as l th and $l-1$ th layer)

$$\text{time complexity} = O\left(\sum_{l=1}^L N_l^2 \times K_l^2 \times C_{l-1} \times C_l\right)$$

N_l : 第 l 層 輸出 的

Feature Map 邊長, N_0

為原 input shape 邊長

K_l : l 層 Kernel 邊長

C_l : l 層 輸出 的 channel 數

C_0 為原先 input 的 channel 數

$$\text{其中, } N_l = \frac{N_{l-1} - K_l + 2 \times P_l}{S_l} + 1$$

5. (1.5%, each 0.5%) PCA practice: Problem statement: Given 10 samples in 3D space. (1, 2, 3), (4, 8, 5), (3, 12, 9), (1, 8, 5), (5, 14, 2), (7, 4, 1), (9, 8, 9), (3, 8, 1), (11, 5, 6), (10, 11, 7)

a. (1) What are the principal axes?

平均值 = (5.4, 8, 4.8)

減去平均值後 = (-4.4, -6, -1.8), (-1.4, 0, 0.2), (-2.4, 4, 4.2),

(-4.4, 0, 0.2), (-0.4, 6, -2.8), (1.6, -4, -3.8),

(3.6, 0, 4.2), (-2.4, 0, -3.8), (5.6, -3, 1.2), (4.6, 3, 2.2)

原資料的 Covariance matrix =

[[13.37777778 0.55555556 3.64444444]

[0.55555556 13.55555556 3.22222222]

[3.64444444 3.22222222 9.06666667]]

Eigenvalues = 54.72, 116.305, 152.974

Principle axes = Eigenvectors = (0.39985541,

0.33758926, -0.85214385), (-0.67817891, 0.73439013, -

0.02728563), (-0.6165947, -0.58881629, -0.52259579)

b. (2) Compute the principal components for each sample.

將原本的 data 與 eigenvectors 所形成的矩陣

[[[0.39985541, -0.67817891, -0.6165947],

[0.33758926, 0.73439013, -0.58881629],

[-0.85214385, -0.02728563, -0.52259579]]]

內積, 可得:

[[-1.48139761, 0.70874446, -3.36201464],

[0.03941652, 3.02597728, -9.78988804],

```
[ -2.41865723,    6.53257419, -13.61894165],
[ -1.16014972,    5.06051399, -7.94010395],
[  5.02123906,    6.83599606, -12.37159312],
[  3.29720109,   -1.83697744, -7.19402383],
[ -1.36988181,   -0.47405978, -14.96324467],
[  3.0481365 ,    3.81329871, -7.0829102 ],
[  0.97349277,   -3.95173109, -12.86219784],
[  1.74702909,    1.10550298, -16.30109667]] (3)
```

Reconstruction error if reduced to 2D. (Calculate the L2-norm)

將 3d 投影到 2d 的矩陣為

```
P = [[-0.67817891 -0.6165947 ]
      [ 0.73439013 -0.58881629]
      [-0.02728563 -0.52259579]]
```

資料被投影到 2d 後：

```
[[ -1.37323947  7.18658682]
 [  0.94399334  0.75871342]
 [  4.45059025 -3.07034019]
 [  2.97853006  2.60849751]
 [  4.75401212 -1.82299166]
 [ -3.91896138  3.35457763]
 [ -2.55604371 -4.41464321]
 [  1.73131477  3.46569126]
 [ -6.03371503 -2.31359638]
 [ -0.97648096 -5.75249521]]
```

再乘上 P^T 還原回 3d：

```
[[ -3.49990928 -5.24007291 -3.71821029]
 [ -1.10801504  0.24651657 -0.42225789]
 [ -1.12514095  5.07633588  1.48310968]
 [ -3.62836199  0.65147726 -1.44446088]
 [ -2.10002375  4.56470677  0.82297154]
 [  0.58934216 -4.85327652 -1.6461568 ]
 [  4.45550052  0.72228056  2.37681721]
 [ -3.31106801 -0.76919499 -1.85839567]
 [  5.51848951 -3.06881754  1.37370944]]
```

[4.20918683 2.6700449 3.03287366]]

和原本的資料計算 L2-norm，10 筆資料對應的 reconstruction error：

[2.25104047 0.73022635 3.1883001 1.92979259 4.25159619 2.52755823
2.13952468 2.27849363 0.2038499 0.97738622]

10 個 reconstruction error 的總和

20.477768358501798