Homework 3 Report - Image Sentiment Classification

B05901022 電機三 許睿洋

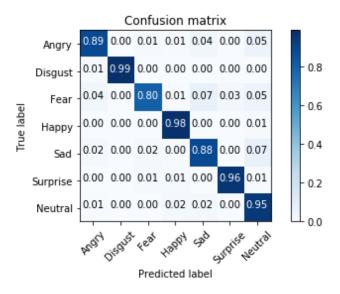
1. (1%) 請說明你實作的 CNN model,其模型架構、訓練過程和準確率為何? 2. (1%) 承上題,請用與上述 CNN 接近的參數量,實做簡單的 DNN model,其模型架構、訓練過程和準確率為何?試與上題結果做比較,並說明你觀察到了什麼?

ver (type) nv2d_1 (Conv2D) tch_normalization_1 (Batch abky_re_lu_1 (LeakyReLU) nv2d_2 (Conv2D) tch_normalization_2 (Batch abky_re_lu_2 (LeakyReLU) nv2d_3 (Conv2D) tch_normalization_3 (Batch abky_re_lu_3 (LeakyReLU) <pre>cooling2d_1 (MaxPooling2</pre>	(None, 48, 48, 64) (None, 48, 48, 64) (None, 48, 48, 64) (None, 48, 48, 64) (None, 48, 48, 64)	Param # 640 256 0 36928 256 0 36928	Layer (type) flatten_1 (flatten) dense_1 (Dense) leaky_re_lu_1 (LeakyReLU) dense_2 (Dense) leaky_re_lu_2 (LeakyReLU) dense_3 (Dense) leaky_re_lu_3 (LeakyReLU) dense_4 (Dense) leaky_re_lu_4 (LeakyReLU) dense_5 (Dense)	Output Shape (None, 2394) (None, 1024) (None, 512) (None, 512) (None, 512) (None, 512)	Paran # 0 2360320 0 1049600 0 1049600 0 524800
tch_normalization_1 (Batch oky_re_lu_1 (LeakyReLU) nv2d_2 (Conv2D) tch_normalization_2 (Batch oky_re_lu_2 (LeakyReLU) nv2d_3 (Conv2D) tch_normalization_3 (Batch oky_re_lu_3 (LeakyReLU)	(None, 48, 48, 64) (None, 48, 48, 64)	256 0 36928 256 0 36928	dense_1 (Dense) Leaky_re_lu_1 (LeakyReLU) dense_2 (Dense) Leaky_re_lu_2 (LeakyReLU) dense_3 (Dense) Leaky_re_lu_3 (LeakyReLU) dense_4 (Dense) Leaky_re_lu_4 (LeakyReLU)	(None, 1024) (None, 512) (None, 512)	2360326 0 1049600 0 1049600 0
oky_re_lu_1 (LeakyReLU) nv2d_2 (Conv2D) tch_normalization_2 (Batch oky_re_lu_2 (LeakyReLU) nv2d_3 (Conv2D) tch_normalization_3 (Batch oky_re_lu_3 (LeakyReLU)	(None, 48, 48, 64) (None, 48, 48, 64)	0 36928 256 0 36928 256	leaky_rc_lu_1 (LeakyReLU) dense_2 (Dense) leaky_re_lu_2 (LeakyReLU) dense_3 (Dense) leaky_re_lu_3 (LeakyReLU) dense_4 (Dense) leaky_re_lu_4 (LeakyReLU)	(None, 1024) (None, 1024) (None, 1024) (None, 1024) (None, 1024) (None, 1024) (None, 512)	0 1049600 0 1049600 0 524800
nvzd_2 (Conv2D) tch_normalization_2 (Batch sky_re_lu_2 (LeakyReLU) nv2d_3 (Conv2D) tch_normalization_3 (Batch sky_re_lu_3 (LeakyReLU)	(None, 48, 48, 64) (None, 48, 48, 64) (None, 48, 48, 64) (None, 48, 48, 64) (None, 48, 48, 64)	36928 256 0 36928 256	leaky_re_lu_2 (LeakyReLU) dense_3 (Dense) leaky_re_lu_3 (LeakyReLU) dense_4 (Dense) leaky_re_lu_4 (LeakyReLU)	(None, 1024) (None, 1024) (None, 1024) (None, 512) (None, 512)	0 1049600 0 524800
tch_normalization_2 (Batch bky_re_lu_2 (LeakyReLU) bv2d_3 (Conv2D) tch_normalization_3 (Batch bky_re_lu_3 (LeakyReLU)	(None, 48, 48, 64) (None, 48, 48, 64) (None, 48, 48, 64) (None, 48, 48, 64)	256 0 36928 256	dense_3 (Dense) leaky_re_lu_3 (LeakyReLU) dense_4 (Dense) leaky_re_lu_4 (LeakyReLU)	(None, 1024) (None, 1024) (None, 512) (None, 512)	1049600 0 524800
oky_re_lu_2 (LeakyReLU) nv2d_3 (Conv2D) cch_normallzation_3 (Batch sky_re_lu_3 (LeakyReLU)	(None, 48, 48, 64) (None, 48, 48, 64) (None, 48, 48, 64)	36928	leaky_re_lu_3 (LeakyReLU) dense_4 (Dense) leaky_re_lu_4 (LeakyReLU)	(None, 1824) (None, 512) (None, 512)	6 524800
nv2d_3 (Conv2D) tch_normalization_3 (Batch aky_re_lu_3 (LeakyReLU)	(None, 48, 48, 64) (None, 48, 48, 64)	36928 256	dense_4 (Dense) leaky_re_lu_4 (LeakyReLU)	(None, 512)	524800
ctch_normalization_3 (Batch sky_re_lu_3 (LeakyReLU)	(None, 48, 48, 64)	256	leaky_re_lu_4 (LeakyReLU)	(None, 512)	No. of Concession, Name of Street, or other teams.
aky_re_lu_3 (LeakyReLU)			dense_5 (Dense)	(None, 512)	
	(None, 48, 48, 64)			***************************************	262656
<pre><_pooling2d_1 (MaxPooling2</pre>		0	leaky_re_lu_5 (LeakyReLU)	(None, 512)	0
	(None, 24, 24, 64)	0	dense_6 (Dense)	(None, 512)	262656
opout_1 (Dropout)	(None, 24, 24, 64)	<u>0</u>	<pre>(leaky_re_lu_6 (LeakyReLU) dense_7 (Dense)</pre>	(None, 512) (None, 512)	262656
nv2d_4 (Conv2D)	(None, 24, 24, 128)	73856	leaky_re_lu_7 (LeakyReLU)	(None, 512)	0
tch_normalization_4 (Batch		512	dense_8 (Dense)	(None, 256)	131328
			leaky_re_lu_8 (LeakyReLU)	(None, 256)	0
		100		(None, 256)	65792
		March Control			32896
		N. 100 (100 (100 (100 (100 (100 (100 (100			0
			dense_11 (Dense)	(None, 128)	16512
			leaky_re_lu_11 (LeakyReLU)	(None, 128)	0
			dense_12 (Dense)	(None, 64)	8256
		A CONTRACTOR OF THE PARTY OF TH			4160
			A service of the serv		0
			dense_14 (Dense)	(None, 32)	2080
nv2d_7 (Conv2D)	(None, 12, 12, 256)	590080	leaky_re_lu_14 (LeakyReLU)	(None, 32)	0
tch_normalization_7 (Batch	(None, 12, 12, 256)	1024	dense_15 (Dense)	(None, 32)	1056
aky_re_lu_7 (LeakyReLU)	(None, 12, 12, 256)	0	A CONTRACTOR OF THE PARTY OF TH		0
			Annual Control of the control of the		1056
					528
					0
nse_1 (Dense)	(None, 512)	4719104	dense_18 (Dense)	(None, 16)	272
tch_normalization_8 (Batch	(None, 512)	2048			0
aky_re_lu_8 (LeakyReLU)	(None, 512)	0			272
opout_4 (Dropout)	(None, 512)	0			136
nse_2 (Dense)	(None, 256)	131328	and the second s		0
tch_normalization_9 (Batch	(None, 256)	1024	dense_21 (Dense)	(None, 8)	72
aky_re_lu_9 (LeakyReLU)	(None, 256)	0	leaky_re_lu_21 (LeakyReLU)	(None, 8)	0
opout_5 (Dropout)	(None, 256)	0	dense_22 (Dense)	(None, 8)	72
nse_3 (Dense)	(None, 7)	1799	And the second s		0
======================================				(None, 7)	63
ainable params: 6,036,871			Trainable params: 6,036,839 Non-trainable params: 0		
	aky_re_lu_4 (LeakyReLU) avzd_5 (Conv2D) acch_normalization_5 (Batch aky_re_lu_5 (LeakyReLU) appout_2 (Dropout) avzd_6 (Conv2D) acch_normalization_6 (Batch aky_re_lu_6 (LeakyReLU) avzd_7 (Conv2D) acch_normalization_7 (Batch aky_re_lu_7 (LeakyReLU) appout_3 (Dropout) atten_1 (Flatten) ase_1 (Dense) acch_normalization_8 (Batch aky_re_lu_8 (LeakyReLU) appout_4 (Dropout) ase_2 (Dense) acch_normalization_9 (Batch aky_re_lu_9 (LeakyReLU) ase_2 (Dense) acch_normalization_9 (Batch aky_re_lu_9 (LeakyReLU) ase_2 (Dense) ase_3 (Dense)	aky_re_lu_4 (LeakyReLU) (None, 24, 24, 128) avzd_5 (ConvZD) (None, 24, 24, 128) avzd_5 (ConvZD) (None, 24, 24, 128) acch_normalization_5 (Batch (None, 24, 24, 128) (None, 24, 24, 128) aky_re_lu_5 (LeakyReLU) (None, 12, 12, 128) appout_2 (Dropout) (None, 12, 12, 128) avzd_6 (ConvZD) (None, 12, 12, 256) acch_normalization_6 (Batch (None, 12, 12, 256) acch_normalization_7 (Batch (None, 12, 12, 256) acch_normalization_7 (Batch (None, 12, 12, 256) acch_normalization_7 (None, 6, 6, 256) acch_normalization_8 (None, 6, 6, 256) acch_normalization_8 (Batch (None, 512) acch_normalization_8 (Batch (None, 512) acch_normalization_9 (None, 512) acch_normalization_9 (Rone, 512) acch_normalization_9 (Rone, 256) acch_normalization_9 (Rone, 256) <td>aky_re_lu_4 (LeakyReLU) (None, 24, 24, 128) 0 avzd_5 (Conv2D) (None, 24, 24, 128) 147584 acch_normalization_5 (Batch (None, 24, 24, 128) 512 aky_re_lu_5 (LeakyReLU) (None, 24, 24, 128) 0 apout_2 (Dropout) (None, 12, 12, 128) 0 avzd_6 (Conv2D) (None, 12, 12, 128) 0 avzd_6 (Conv2D) (None, 12, 12, 128) 0 avzd_6 (Conv2D) (None, 12, 12, 256) 295168 acch_normalization_6 (Batch (None, 12, 12, 256) 1024 aky_re_lu_6 (LeakyReLU) (None, 12, 12, 256) 590080 acch_normalization_7 (Batch (None, 12, 12, 256) 1024 aky_re_lu_7 (LeakyReLU) (None, 12, 12, 256) 0 acch_normalization_7 (Batch (None, 12, 12, 256) 1024 aky_re_lu_7 (LeakyReLU) (None, 6, 6, 256) 0 apout_3 (Dropout) (None, 6, 6, 256) 0 atten_1 (Flatten) (None, 9216) 0 atten_1 (Flatten) (None, 512) 2048 aky_re_lu_8 (LeakyReLU) (None, 512) 2048 aky_re_lu_8 (LeakyReLU) (None, 512) 0 apout_4 (Dropout) (None, 512) 0 ase_2 (Dense) (None, 256) 131328 acch_normalization_9 (Batch (None, 256) 1024 aky_re_lu_9 (LeakyReLU) (None, 256) 0 ase_3 (Dense) (None, 7) 1799 aral params: 6,046,3277 atal params: 6,046,3277</td> <td> lesky_re_lu_8 (leskyReLU) lesky_re_lu_8 (leskyReLU) lesky_re_lu_8 (leskyReLU) lesky_re_lu_9 (leskyReLU) </td> <td> Lesky_re_lu_4 (LeakyReLU) (None, 24, 24, 128) 0 Lesky_re_lu_9 (LeskyReLU) (None, 256) </td>	aky_re_lu_4 (LeakyReLU) (None, 24, 24, 128) 0 avzd_5 (Conv2D) (None, 24, 24, 128) 147584 acch_normalization_5 (Batch (None, 24, 24, 128) 512 aky_re_lu_5 (LeakyReLU) (None, 24, 24, 128) 0 apout_2 (Dropout) (None, 12, 12, 128) 0 avzd_6 (Conv2D) (None, 12, 12, 128) 0 avzd_6 (Conv2D) (None, 12, 12, 128) 0 avzd_6 (Conv2D) (None, 12, 12, 256) 295168 acch_normalization_6 (Batch (None, 12, 12, 256) 1024 aky_re_lu_6 (LeakyReLU) (None, 12, 12, 256) 590080 acch_normalization_7 (Batch (None, 12, 12, 256) 1024 aky_re_lu_7 (LeakyReLU) (None, 12, 12, 256) 0 acch_normalization_7 (Batch (None, 12, 12, 256) 1024 aky_re_lu_7 (LeakyReLU) (None, 6, 6, 256) 0 apout_3 (Dropout) (None, 6, 6, 256) 0 atten_1 (Flatten) (None, 9216) 0 atten_1 (Flatten) (None, 512) 2048 aky_re_lu_8 (LeakyReLU) (None, 512) 2048 aky_re_lu_8 (LeakyReLU) (None, 512) 0 apout_4 (Dropout) (None, 512) 0 ase_2 (Dense) (None, 256) 131328 acch_normalization_9 (Batch (None, 256) 1024 aky_re_lu_9 (LeakyReLU) (None, 256) 0 ase_3 (Dense) (None, 7) 1799 aral params: 6,046,3277 atal params: 6,046,3277	lesky_re_lu_8 (leskyReLU) lesky_re_lu_8 (leskyReLU) lesky_re_lu_8 (leskyReLU) lesky_re_lu_9 (leskyReLU)	Lesky_re_lu_4 (LeakyReLU) (None, 24, 24, 128) 0 Lesky_re_lu_9 (LeskyReLU) (None, 256)

CNN 說明& 比較 結果 我的 CNN 架構採用類似 VGG16 多層卷積層後才連接池化層的方式,共疊了三個大層後再接兩層 Fully Connected,並以 Softmax 輸出。統一使用 LeakyReLU(alpha=0.3)當 activation function、Dropout=0.3,並加入 BatchNormalization 改善。

使用相同參數量的 DNN 模型很顯然會在前幾個 epoch 便嚴重 overfit 了。雖然本來使用 fully connected 就會很容易 overfit,但是相同數量的參數下 CNN 還有相當漂亮的結果,因此可以得知使用圖片的特性來訓練一個模型在處理影響辨識上相當重要。

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]



容易搞混的 class:

- (1) Fear⇔Sad, Neutral, Angry
- (2) Sad⇔Neutral
- (3) Angry⇔Sad, Neutral

Neutral(中性)的 label 針對自己的預測準確率很高,但其他的 label(Fear, Sad, Angry)卻很容易預測到 Neutral 上。在所有的 label 中,Fear 的準確率最低,Disgust 跟 Happy 的準確率最高。

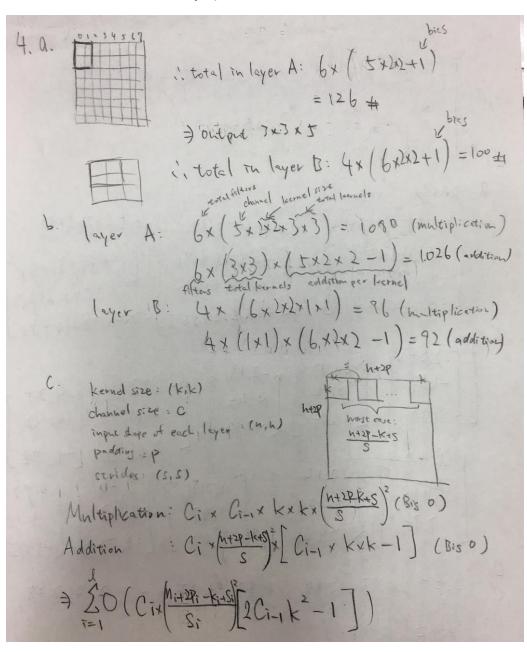
4. (1.5%, each 0.5%)CNN time/space complexity:

For a. b. Given a CNN model as

And for the c. given the parameter as:

kernel size = (k, k); channel size = c; input shape of each layer = (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)
- b. How many multiplications/additions are needed for a forward pass (each layer).
- c. What is the time complexity of convolutional neural networks? (note: you must use big-O upper bound, and there are I (lower case of L) layer, you can use CI, CI-1as Ith and I-1th layer)



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)

- (1) What are the principal axes?
- (2) Compute the principal components for each sample.
- (3) Reconstruction error if reduced to 2D. (Calculate the L2-norm)

Covariance matrix

$$\begin{bmatrix}
12.04 & 0.5 & 3.28 \\
0.5 & 12.2 & 2.9 \\
3.29 & 2.9 & 9.16
\end{bmatrix}$$

a. Principle axes:
$$\lambda_{=15,2974934} \quad \lambda_{=11,13052369} \quad \lambda_{3} = 5.49203291$$

$$\begin{bmatrix}
-0.6165.747 \\
-0.51999629 \\
-0.73431013
\end{bmatrix}
\begin{bmatrix}
0.319735497 \\
0.33959996
\end{bmatrix}$$
b.
$$\begin{bmatrix}
-2.25 \\
-1.37 \\
7.19 \\
7.19
\end{bmatrix}
\begin{bmatrix}
-0.073 \\
0.94 \\
7.19
\end{bmatrix}
\begin{bmatrix}
-1.19 \\
4.45 \\
-2.56
\end{bmatrix}
\begin{bmatrix}
-1.93 \\
2.61
\end{bmatrix}
\begin{bmatrix}
4.25 \\
4.07
\end{bmatrix}
\begin{bmatrix}
-1.92 \\
4.07
\end{bmatrix}
\begin{bmatrix}
-2.14 \\
-2.56
\end{bmatrix}
\begin{bmatrix}
-2.14 \\
-2.57
\end{bmatrix}
\begin{bmatrix}
-2.14 \\$$