

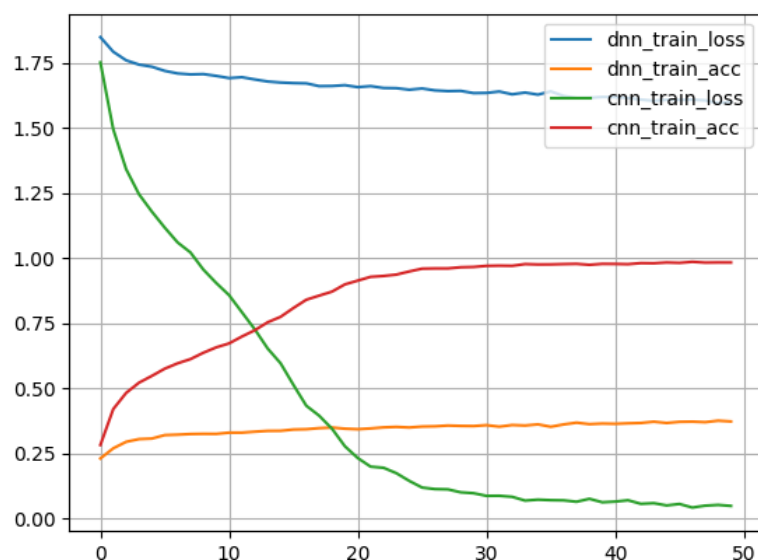
學號：B05901170 系級：電機三 姓名：陳柏志

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: none)

	CNN Model	DNN Model
模型架構	<pre> ===== Conv2d-1 [-1, 64, 48, 48] BatchNorm2d-2 [-1, 64, 48, 48] 128 LeakyReLU-3 [-1, 64, 48, 48] 0 Conv2d-4 [-1, 64, 48, 48] 36,928 BatchNorm2d-5 [-1, 64, 48, 48] 128 LeakyReLU-6 [-1, 64, 48, 48] 0 MaxPool2d-7 [-1, 64, 24, 24] 0 Conv2d-8 [-1, 64, 24, 24] 36,928 BatchNorm2d-9 [-1, 64, 24, 24] 128 LeakyReLU-10 [-1, 64, 24, 24] 0 Conv2d-11 [-1, 64, 24, 24] 36,928 BatchNorm2d-12 [-1, 64, 24, 24] 128 LeakyReLU-13 [-1, 64, 24, 24] 0 MaxPool2d-14 [-1, 64, 12, 12] 0 Conv2d-15 [-1, 128, 12, 12] 73,856 BatchNorm2d-16 [-1, 128, 12, 12] 256 LeakyReLU-17 [-1, 128, 12, 12] 0 Conv2d-18 [-1, 128, 12, 12] 147,584 BatchNorm2d-19 [-1, 128, 12, 12] 256 LeakyReLU-20 [-1, 128, 12, 12] 0 MaxPool2d-21 [-1, 128, 6, 6] 0 Linear-22 [-1, 512] 2,359,808 LeakyReLU-23 [-1, 512] 0 Dropout-24 [-1, 512] 0 Linear-25 [-1, 512] 262,656 LeakyReLU-26 [-1, 512] 0 Dropout-27 [-1, 512] 0 Linear-28 [-1, 7] 3,591 ===== Total params: 2,959,943 </pre>	<pre> ===== Linear-1 [-1, 1024] LeakyReLU-2 [-1, 1024] Dropout-3 [-1, 1024] Linear-4 [-1, 1024] LeakyReLU-5 [-1, 1024] Dropout-6 [-1, 1024] Linear-7 [-1, 7] ===== Total params: 3,417,095 </pre>
訓練參數	<p>Batch size = 256, Epoch = 50, Shuffle = True, No Data Augmentation, Optimizer = Adam with learning rate = 0.001</p>	<p>Batch size = 256, Epoch = 50, Shuffle = True, No Data Augmentation, Optimizer = Adam with learning rate = 0.001</p>
準確率	<p>Public Score: 0.60406 Private Score: 0.58651</p>	<p>Public Score: 0.38422 Private Score: 0.39091</p>

觀察：即使有經過調整使兩模型參數量接近，甚至使 DNN 模型參數在略多於 CNN 模型。仍能明顯可以看出 CNN 模型的表現較 DNN 優異許多。而有趣的是 DNN 模型的 private score 比 public score 來得高。

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: none)



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: none)

3.1. data normalization 部份我將(0, 255)的資料 map 成(0, 1)，再進行訓練。下表為實作結果。

	CNN with data normalization	CNN without data normalization
Public score	0.60406	0.60128
Private score	0.58651	0.57536

可以看出施行 data normalization 對於訓練有些許改善作用。但在此的差異相當小，也可能只是隨機成份造成如此的差異。

3.2. data augmentation 部份我使用 torchvision.transform 來生成 8 倍的額外資料。生成方式包含將圖片隨機旋轉 (in 30 degree)，隨機縮放(range(0.7, 1.0))，及 random crop。再將這些額外資料加入 train data 中進行訓練。下表為實作結果。

	CNN with data augmentation	CNN without data augmentation
Public score	0.68013	0.60406
Private score	0.67261	0.58651

可以看出 data augmentation 對於訓練結果有相當大的影響，額外的資料對於訓練相當有幫助。

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: none)

這邊為了取得正確 label，我使用 train data 來做辨識並取得結果如下表。其中數字單位為機率。

\Result Label \	0	1	2	3	4	5	6
0	0.8821	0.0138	0.0186	0.0098	0.0360	0.0054	0.0256
1	0.0025	0.9335	0.0012	0.0003	0.0010	0.0009	0.0008
2	0.0398	0.0069	0.8243	0.0065	0.0534	0.0356	0.0282
3	0.0075	0.0000	0.0041	0.9771	0.0058	0.0082	0.0129
4	0.0328	0.0023	0.0410	0.0098	0.8689	0.0050	0.0495
5	0.0070	0.0069	0.0151	0.0058	0.0029	0.9461	0.0044
6	0.0193	0.0046	0.0098	0.0290	0.0366	0.0069	0.8939

由表中可以看出相對較容易混淆的種類有[0(生氣), 4(難過)] 及 [2(恐懼), 4(難過)] 兩組。