EEML HW3

B06702064 會計五 林聖硯

模型連結

https://drive.google.com/file/d/1P1P0Zp6fzIlkJP9-S0t4jy9eQs8oaVQd/view?usp=sharing
(https://drive.google.com/file/d/1P1P0Zp6fzIlkJP9-S0t4jy9eQs8oaVQd/view?usp=sharing)

npy files連結

https://drive.google.com/drive/folders/1OxWKil6soXjIU4mhbnMydkL6csbjbviv?usp=sharing
(https://drive.google.com/drive/folders/1OxWKil6soXjIU4mhbnMydkL6csbjbviv?usp=sharing)

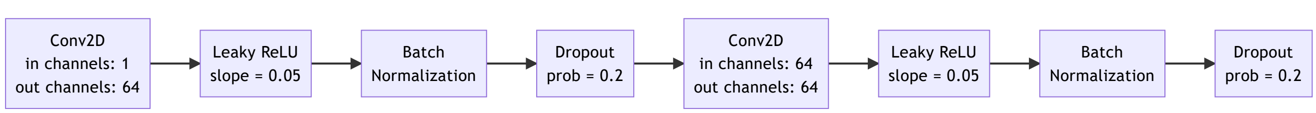
助教非常不好意思，我原本都是讀取自己存起來的npy files來訓練模型，比每次重新從資料夾裡面讀取圖片有效率許多，但重新改回助教讀取圖片的code之後acc就瘋狂下降，而且testing時間要耗時30分鐘，我在最後上傳作業前才發現(剩下半個小時就要交作業)，實在是來不及修改，懇請助教能給我一點分數...。另外，我明明已經設定過seed，但是將model讀回來重做作時，acc卻掉了10%，不知道這部分到底是出了什麼問題QQ，真的非常抱歉

兩種讀取檔案並且testing的方式詳見readme

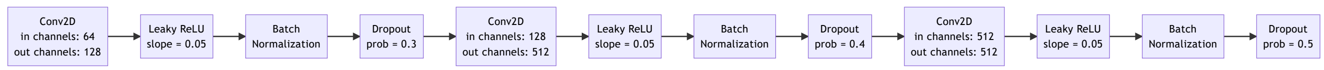
1. (1%) 請以block diagram或是文字的方式說明這次表現最好的model使用哪些layer module(如 Conv/Linear 和各類 normalization layer) 及連接方式(如一般forward 或是使用 skip/residual connection)，並概念性逐項說明選用該 layer module 的理由。

我最一開始的model只有使用使用兩層conv+pooling，得到的training結果為99%、validation的結果為50%左右，很明顯地overfit。後來加上batch normalization以及dropout之後，模型反而underfit dataset，所以我最後又再加上了幾層convolution layer才得到最好的結果。我模型的block diagram如下：

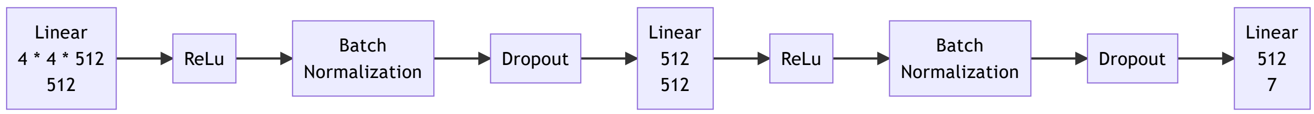
CNN (前兩層)



CNN (後三層)



FNN



我的CNN主要分成兩個部分

- 第一層以及第二層的convolutional layer主要抓出比較general的資訊(大塊的資訊)，所以這兩層的kernel size為5x5。而output channels比較小的原因是，比較大size的資訊對判斷表情沒有太大的幫助。這裡的dropout rate也設定的比較小，因為這邊的模型也比較簡單，不容易overfit。這裡的兩個block中都沒有放上maxpooling，原因是我希望這兩層的模型只是對大塊的資訊做一些feature transformation，maxpooling反而會導致模型細部的資訊喪失。
- 第三、第四以及第五層的convolutional layer主要是拿來抓出較為細部的資訊(我的assumption是細部的資訊才有助於判斷表情)，所以這裡kernel設定3x3，深度也設定為三層convolutional layer讓模型能夠多抓出一點局部的資訊。但模型複雜度上升之後，相對地也需要調高dropout rate來防止模型overfitting。
- 這五層都有放入batch normalization主要是為了防止internal covariate shift，也同時能加速模型收斂的速度。
- 關於各種convolution layer裡面各層的擺放順序我主要是參考以下兩篇paper的觀點，設定為conv -> activation function -> batch norm -> maxpool -> dropout -> conv
  - https://arxiv.org/pdf/1502.03167.pdf (https://arxiv.org/pdf/1502.03167.pdf)
  - https://www.cs.toronto.edu/~hinton/absps/JMLRdropout.pdf (https://www.cs.toronto.edu/~hinton/absps/JMLRdropout.pdf)

2. (1%) 嘗試使用 augmentation/early-stopping/ensemble 三種訓練 trick 中的兩種，說明實作細節並比較有無該 trick 對結果表現的影響(validation 或是 testing 擇一即可)。

我這邊使用了data augmentaion以及early stopping兩種技巧來幫助模型訓練

- data augmentation主要是為了讓模型generalize的能力能夠更強，我使用了pytorch裡面torchvision的transforms模組，並且對**training dataset**作了以下兩個transform
  - RandomVerticalFlip
  - RandomRotation(degrees = (-20, 20))
  - Normalize(mean=0.508156, std=0.264441)

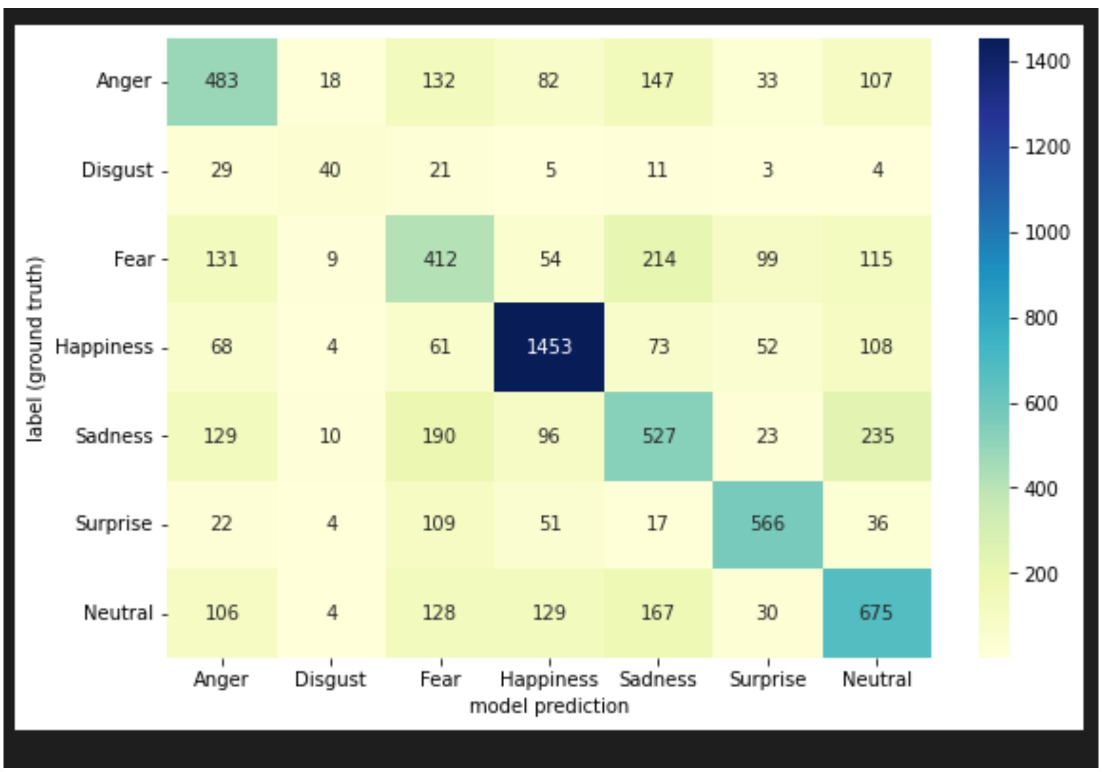
而關於validation以及testing set，我只有分別對他們使用資料集的mean以及std進行normalization(當然也有做在training set上)

- early stopping
    - 使用validation loss來決定要不要early stopping
    - patience設定為10(i.e.如果validation loss在10個epoch內沒有上升，則停止模型訓練)

訓練技巧	training loss	validation loss
不做data augmentation以及early stopping	99.99%	50%
只有 data augmentation	99.99%	62%
只有 early stopping	85.12%	60.73%
data augmentation + early stopping	71.82%	64.71%

3. (1%) 畫出 confusion matrix 分析哪些類別的圖片容易使 model 搞混，並簡單說明。
   (ref: https://en.wikipedia.org/wiki/Confusion_matrix (https://en.wikipedia.org/wiki/Confusion_matrix))

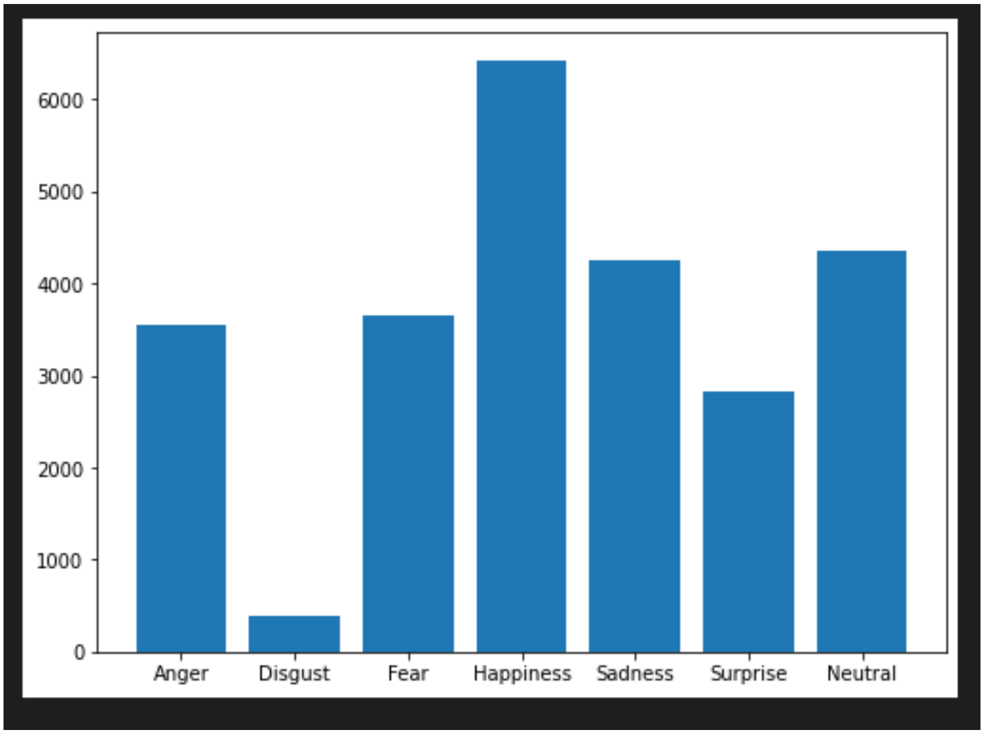
從confusion matrix來看，可以看出模型在Disgust類別的表現較差，原因是disgust在training set的比例明顯比其他類別的圖片低很多。



4. (1%) 請統計訓練資料中不同類別的數量比例，並說明：
   對 testing 或是 validation 來說，不針對特定類別，直接選擇機率最大的類別會是最好的結果嗎？針對上述內容，是否存在更好的方式來提升表現？例如設置不同條件來選擇預測結果/變更訓練資料抽樣的方式，或是直接回答「否」(但需要給出支持你論點的論述)

資料中不同類別的圖片比率如下：

	Category	Ratio
0	Anger	13.878215
1	Disgust	1.557794
2	Fear	14.317859
3	Happiness	25.184339
4	Sadness	16.758403
5	Surprise	11.150344
6	Neutral	17.153045



直接選擇機率最大的類別不會是最好的結果，以下我做了幾個我認為能夠改善模型效果的方式。

- 在切train/valid dataset時，進行stratify(也就是讓各類別的data能均勻散布在training以及validation set裡面)
  - 有使得模型在validation set上的結果提升2%
- 在計算cross entropy loss時將每一個class的ratio也考慮進去loss function，我使用了以下的兩種考慮方式，但都導致loss的下降非常不穩定也沒辦法收斂到最好的結果
  - 1 / # of observations in each class
  - inverse ratio of each class
- 不直接使用argmax，而是先看預測的機率有沒有高於原本class的機率，再從有高於的那些class中取argmax
  - 這個方法我是最後才想到的，但來不及做測試，不知道會不會有用

Math

Problem 1

1、

image data: $(B, W, H, \text{input\_channels})$

from the formula of CONV2D in pytorch, we know that

$$W_{out} \; (H_{out}) = \left\lfloor \frac{W_{in} \, (H_{out}) + 2 \times \text{padding} - \text{dilation} \times (\text{kernel\_size} - 1) - 1}{\text{stride}} + 1 \right\rfloor$$

$$\Rightarrow W_{out} = \left\lfloor \frac{W + 2p_1 - k_1}{s_1} + 1 \right\rfloor , \quad H_{out} = \left\lfloor \frac{H + 2p_2 - k_2}{s_2} + 1 \right\rfloor$$

$\Rightarrow$ image size after conv. layer: $(B, W_{out}, H_{out}, \text{output\_channels})$

Problem 2

2.

① loss function: $\ell(y_i)$ ← output after BN

② $y_i = \gamma \hat{x}_i + \beta$ (γ, β are scalers) $(y = y(\gamma, \hat{X}, \beta))$

③ $\hat{x}_i = \dfrac{x_i - u_B}{\sqrt{\sigma_B^2 + \epsilon}}$, $u_B = \frac{1}{m} \sum_{i=1}^{m} x_i$, $\sigma_B^2 = \frac{1}{m} \sum_{i=1}^{m} (x_i - u_B)^2$ $(\hat{X} = \hat{X}(x_i, u_B, \sigma_B^2))$

(mini-batch mean & std)

Calculate all derivatives in ②

$$\frac{\partial \ell}{\partial \gamma} = \sum_{i=1}^{m} \frac{\partial \ell}{\partial y_i} \cdot \hat{x}_i , \quad \frac{\partial \ell}{\partial \hat{x}_i} = \frac{\partial \ell}{\partial y_i} \cdot \gamma , \quad \frac{\partial \ell}{\partial \beta} = \sum_{i=1}^{m} \frac{\partial \ell}{\partial y_i} . \; \#$$

Calculate derivatives in ③

$$\frac{\partial \ell}{\partial \sigma_B^2} = \sum_{i=1}^{m} \frac{\partial \ell}{\partial \hat{x}_i} \cdot \left[-\frac{1}{2} (x_i - u_B) (\sigma_B^2 + \epsilon)^{-\frac{3}{2}} \right] \; \#$$

$$\frac{\partial \ell}{\partial u_B} = \frac{\partial \ell}{\partial \hat{x}_i} \left(\frac{\partial \hat{x}_i}{\partial u_B} + \frac{\partial \hat{x}_i}{\partial \sigma_B^2} \cdot \frac{\partial \sigma_B^2}{\partial u_B} \right)$$

$\hat{x}_i \leftarrow u_B$.

$\hat{x}_i \leftarrow \sigma_B^2 \leftarrow u_B$.

$$= \sum_{i=1}^{m} \frac{\partial \ell}{\partial \hat{x}_i} \left[\frac{-1}{\sqrt{\sigma_B^2 + \epsilon}} + \left(-\frac{1}{2} (x_i - u_B) (\sigma_B^2 + \epsilon)^{-\frac{3}{2}} \cdot \left(-\frac{2}{m} \sum_{i=1}^{m} (x_i - u_B) \right) \right) \right]$$

$\frac{\partial \ell}{\partial \sigma_B^2}$

Problem3

3、

class i

cross-entropy: $L(y,\hat{y}) = -\sum_i y_i \log \hat{y}_i$

ground truth $(0,0,0,\cdots,1,0,0)$

cross-entropy at t^{th} step $= L(y_t,\hat{y}_t) = -y_t \log y_t$

(class i at step t)

$$\text{softmax}(z_t) = \frac{e^{z_t}}{\sum_i e^{z_i}} = \hat{y}.$$

∂

$$\frac{\partial L_{z_t}}{\partial z_t} = \frac{\partial}{\partial z_t}(-y_t \log \hat{y}_t) = -y_t \frac{\partial}{\partial z_t}(\log \hat{y}_t)$$

$$= -y_t \cdot \frac{1}{\hat{y}_t} \cdot \frac{\partial}{\partial z_t}(\text{softmax}(z_t))$$

Compute $\frac{\partial}{\partial z_t}\left(\frac{e^{z_t}}{\sum_i e^{z_i}}\right) = \frac{e^{z_t}\sum_i e^{z_i} - e^{2z_t}}{\left(\sum_i e^{z_i}\right)^2} = \frac{e^{z_t}}{\sum_i e^{z_i}} \cdot \frac{\sum_i e^{z_i} - e^{z_t}}{\sum_i e^{z_i}} = \hat{y}_t(1-\hat{y}_t)$

$$\therefore \frac{\partial L_{z_t}}{\partial z_t} = -y_t \cdot \frac{1}{\hat{y}_t} \cdot \hat{y}_t(1-\hat{y}_t)$$

$$= y_t(\hat{y}_t - 1)$$

$\because y_t = 1$ (ground truth) $= \hat{y}_t - y_t$

Problem 4

(a)

$$m^t = \beta_1 m^{t-1} + (1-\beta_1)\cdot g^t$$

$$= \beta_1 \left[\beta_1 m^{t-2} + (1-\beta_1) g^{t-1}\right] + (1-\beta_1) g^t$$

$$= (\beta_1)^2 m^{t-2} + \beta_1 (1-\beta_1) g^{t-1} + (1-\beta_1) g^t$$

$$= (\beta_1)^2 \left[\beta_1 m^{t-3} + (1-\beta_1) g^{t-2}\right] + \beta_1 (1-\beta_1) g^{t-1} + (1-\beta_1) g^t$$

$$= (\beta_1)^3 m^{t-3} + (\beta_1)^2 (1-\beta_1) g^{t-2} + \beta_1 (1-\beta_1) g^{t-1} + (1-\beta_1) g^t$$

$$= \cdots = \sum_{i=1}^{t} (\beta_1)^t m_0 + \sum_{i=1}^{t} (\beta_1)^{t-i} (1-\beta_1) g^i$$

$$= \underbrace{(1-\beta_1)}_{A} \sum_{i=1}^{t} \underbrace{(\beta_1)^{t-i}}_{B.} g^i$$

$$V_t = \beta_2 \cdot V^{t-1} + (1-\beta_2)\cdot (g^t)^2$$

$$= \beta_2 \left[\beta_2 V^{t-2} + (1-\beta_2)(g^{t-1})^2\right] + (1-\beta_2)(g^t)^2$$

$$= (\beta_2)^2 V^{t-2} + \beta_2 (1-\beta_2)(g^{t-1})^2 + (1-\beta_2)(g^t)^2$$

$$= \cdots = \sum_{i=0}^{t} (\beta_2)^t V^0 + \sum_{i=1}^{t} (\beta_2)^i (1-\beta_2)(g^{t-i})^2$$

$$= \underbrace{(1-\beta_2)}_{C} \sum_{i=1}^{t} \underbrace{(\beta_2)^{t-i}}_{D.} (g^i)^2.$$

Ans: $\begin{cases} A = 1-\beta_1 \\ B_i = \beta_1^{t-i} \\ C = 1-\beta_2 \\ D_i = \beta_2^{t-i} \end{cases}$

(b)

$$\hat{V}^t = \frac{V^t}{1-\beta_2^t}$$

$$= \frac{(1-\beta_2)\sum_{i=1}^{t}\beta_2^{t-i}\cdot(g^i)^2}{1-\beta_2^t}$$

$$= \frac{\boxed{(1-\beta_2)}\boxed{[(g^t)^2+\beta_2^1(g^{t-1})^2+\beta_2^2(g^{t-2})^2+\cdots+\beta_2^{t-1}(g^1)^2]}}{\boxed{1-\beta_2^t}}.$$

$$\lim_{\beta_2\to 1}\hat{V}^t = \lim_{\beta_2\to 1}\left(\frac{1-\beta_2}{1-\beta_2^t}\right)\cdot\left[(g^t)^2+\beta_2^1(g^{t-1})^2+\cdots+\beta_2^{t-1}(g^1)^2\right] \overset{def}{=} G$$

$$(\beta_2)^0\sum_{i=1}^{t}(g^i)^2 \le G \le \beta_2^{t-1}\sum_{i=1}^{t}(g^i)^2$$

By squeeze thm, $\lim_{\beta_2\to 1} G = \sum_{i=1}^{t}(g^i)^2$

By L'Hôspital Rule, $\lim_{\beta_2\to 1}\frac{1-\beta_2}{1-\beta_2^t} = \lim_{\beta_2\to 1}\frac{-1}{-t\beta_2^{t-1}} = \frac{1}{t}$.

$$= t^{-1}\cdot\sum_{i=1}^{t}(g^i)^2.$$

If $\beta_1 = 0,\ \beta_2\to 1,\ \eta = \eta_0\cdot t^{-\frac{1}{2}}$

$$w^t = w^{t-1} - \frac{\eta_0\cdot t^{-\frac{1}{2}}}{\sqrt{t^{-1}\cdot\sum_{i=1}^{t}(g^i)^2}}g^t$$

$$= w^{t-1} - \frac{\eta_0}{\sqrt{\sum_{i=1}^{t}(g^i)^2}}\cdot g^t$$ which is the form of Adagrad optimizer.