1. DNN 手寫辨識

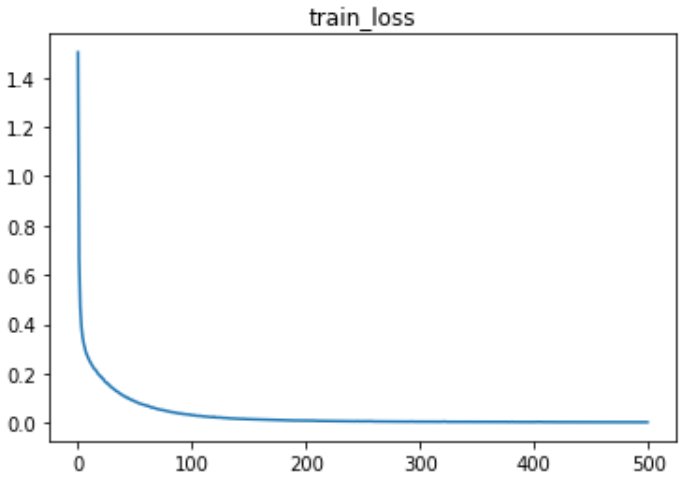
i. 自刻 DNN 分類，使用 loss function 為 cross entropy；error backpropagation algorithm 使用 SGD:

***作業尾末有自刻 DNN 理論推算。

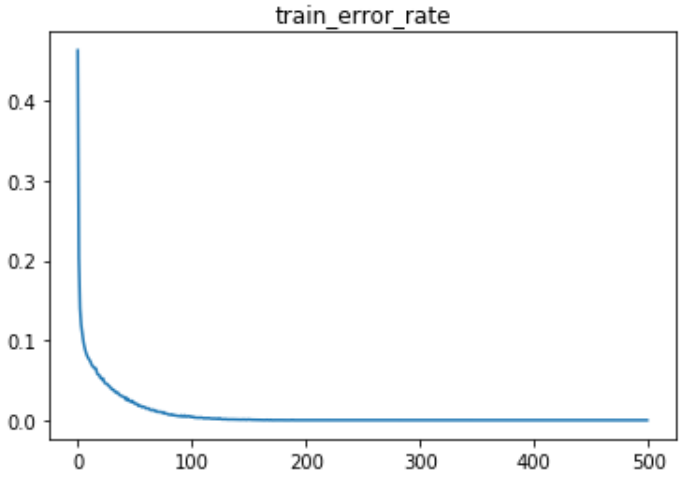
使用一層 hidden layer，該層節點(node)為 60，原本設為 1500，但後來試跑一下發現一個 epoch 要一分多鐘，等到 500 個 epoch 會死掉。所以後來才改為 60 個節點，準確率與 loss 與準確率在前面約莫 10 個 epoch 較 1500 個節點的慢，但後來 loss 與準確率與 1500 個節點的差不多。

以 batch size 為 120 做 500 個 epoch 的迭代，下圖所使用的 loss function 為每筆資料的 cross entropy 取平均、error rate 為 1-accuracy rate。

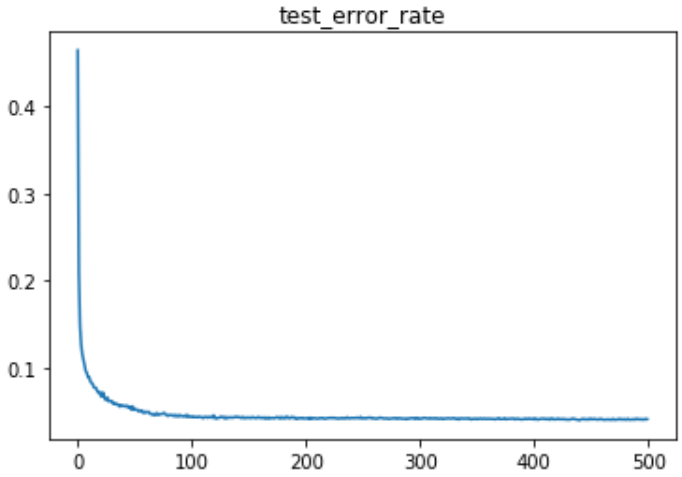
(1) Learning curve



(2) Training error rate



(3) Test error rate

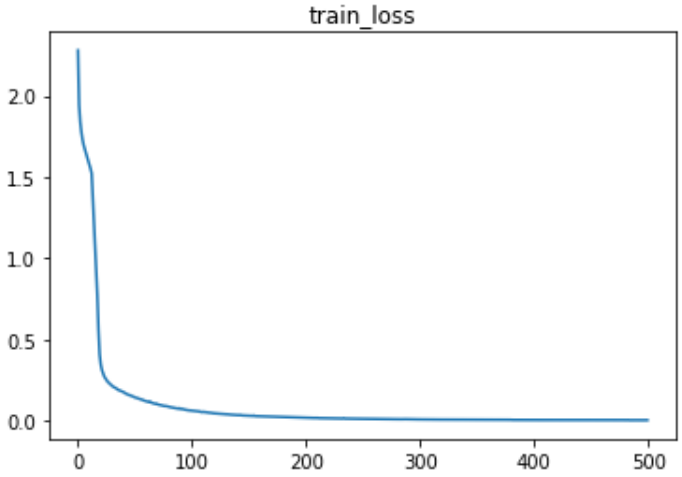


前期約 20 個 epoch 的 train loss 有明顯地下降，說明參數(w, b)在迭代更新過程中能有效降 loss。Loss 與 train error rate 會是正相關的兩個函數(因為 loss 也是衡量 error 的指標)。

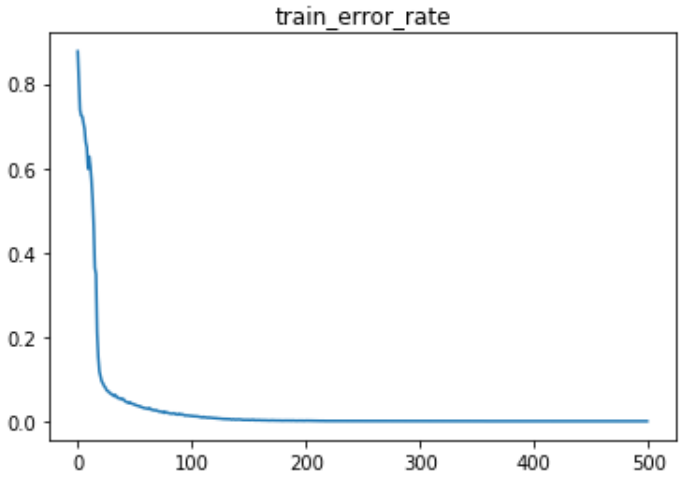
Test error rate 有一點很有趣: 把圖片放大後會發現有些為抖動；但在 train error rate 中沒看到，說明模型可能有 overfitting 的問題。

ii. 第 i 小題是使用 random initialization 作權重的初始值。在這小題中，模型的架構和上個小題一樣，但會以 0 為初始權重，但針對 b(就是 wx+b 之中的 b)參數我們一樣用 random 給值，因為如果 b 設 0，參數根本不會更新。

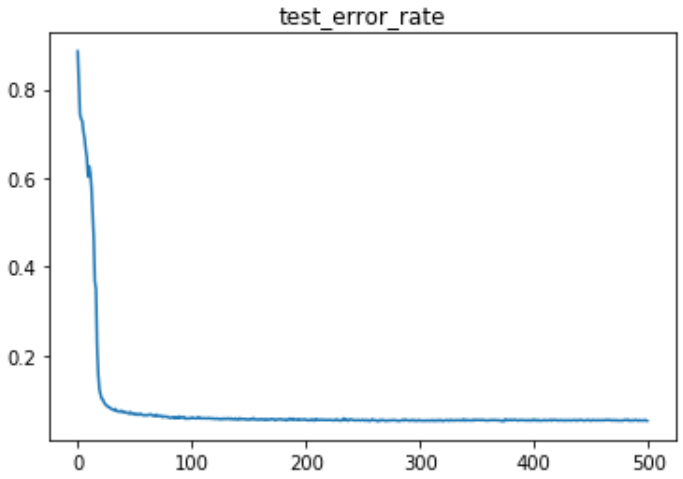
(1) Learning curve



(2) Training error rate



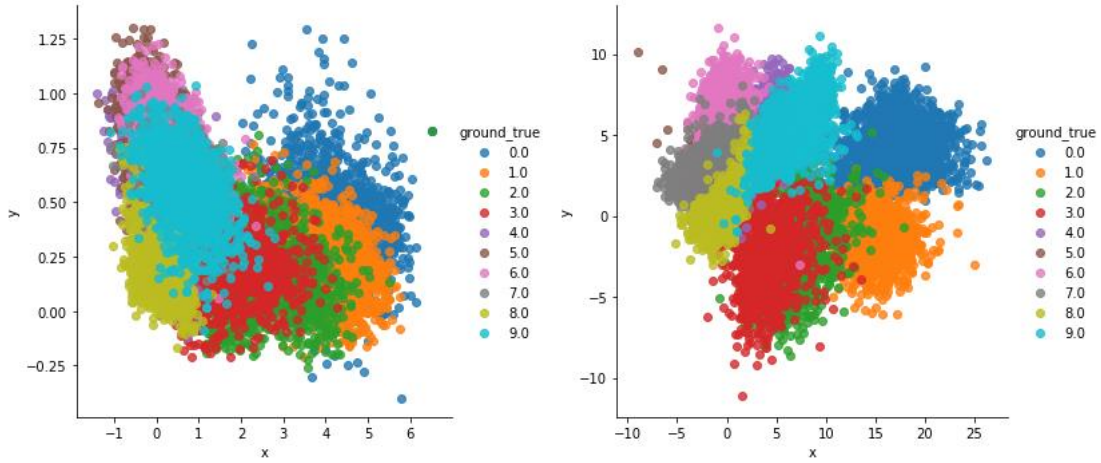
(3) Test error rate



以零為初始的例子當中，後期表現會與差不多，代表模型很穩健，表現不會因為初始值不同而有很大的差別。但前期看得出來會相對不穩定: 可能的猜想是:
因為第一個 epoch 並沒有給權重任何一個方向，也就是完全沒有用到 train data 的資訊。導致模型在第一次迭代時只能根據隨機給定的 b 向量來決定方向，導致給出的方向也是亂選(沒有根據資料的資訊亂取方向)。因而在前期繞了冤枉路。

iii. 把最後面一層 layer 的 node 改成 2: 我的模型為一層 hidden layer，該層節點(node)為 2。做 200 個 epoch。

(1) 它在第一個 epoch 中 train data 的準確率為: 0.1425、test data 的準確率為 0.14597781；第 200 個 epoch 中，train data 的準確率為: 0.781、test data 的準確率為 0.73803745
左下圖為第一個 epoch 根據 hidden layer 中兩個 node 和 ground true 所做的圖、右下圖為第 200 個 epoch 根據 hidden layer 中兩個 node 和 ground true 所做的圖。



可以明顯看出一開始甚至連肉眼都沒辦法分出 10 類、在第 200 個 epoch 時情況會好很多；但仍然有一些類別是重疊的。
列出最後一個 epoch 的 confusion matrix:

	0	1	2	3	4	5	6	7	8	9
0	613	37	2	0	1	0	1	0	0	10
1	24	613	24	0	0	0	0	0	0	0
2	16	33	387	143	0	0	0	0	1	4
3	4	12	134	368	7	3	1	8	52	11
4	1	0	1	17	459	6	23	22	22	100
5	0	0	0	1	16	256	40	79	15	0
6	0	0	0	0	24	65	407	6	0	0
7	0	0	0	9	43	168	22	106	94	7
8	0	0	0	22	27	4	1	30	488	0
9	13	0	5	18	74	1	1	2	4	560

因為到最後 2, 3 重疊情況嚴重，觀察 confusion matrix 也能發現這點。

iv. 根據第 ii 小題最後一個 epoch 迭出來的參數做 test data 的 confusion matrix。

	0	1	2	3	4	5	6	7	8	9
0	657	4	1	1	0	0	1	0	0	0
1	4	653	2	1	0	0	0	0	0	1
2	8	7	534	34	0	0	0	0	1	0
3	1	1	34	539	3	2	7	5	6	2
4	1	0	1	2	593	1	13	1	15	24
5	0	0	0	2	2	391	4	6	1	1
6	1	0	0	1	2	6	482	8	1	1
7	2	0	0	0	4	6	10	419	3	5
8	0	0	0	4	8	1	3	1	555	0
9	1	4	2	1	20	5	2	4	1	638

細看其中有兩點:

(1) 2、3 錯分彼此的比例大: 目測看來，2、3 形狀很像，3 感覺像基於 2 多加一些點；然而 1 個架構看起來很像 2、3，但因為撇的方向不一樣，在圖像資訊上就會有很大的差別。

(2) 4、9 錯分彼此的比例大: 它們兩個目測看起來不像。但主觀猜測是因為節點或是 hidden layer 不夠多、模型不夠複雜，因而導致為了讓其他幾個分類情況好而有所取捨。

2. RGB 圖像辨識: 有沒有戴好口罩

i. 我先把大圖載下來，然後利用 csv 檔裡面的資訊把小圖切下來儲存，針對各個小圖做 resize。

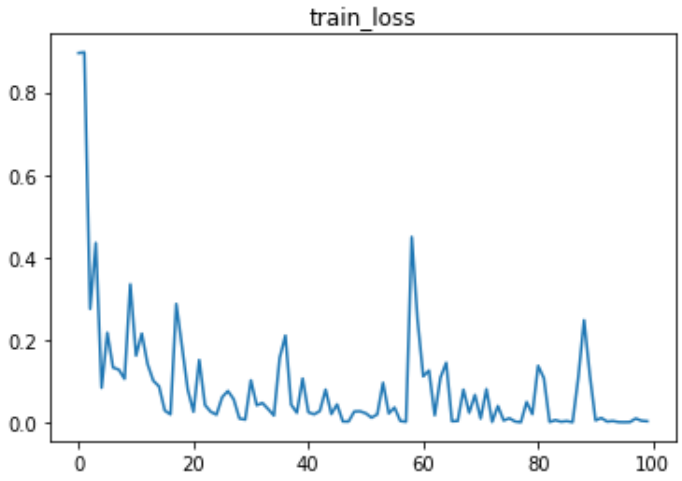
用 opencv 中的 cv2.size: 把圖片壓成 28*28 的樣子。

解釋: 原本大小為 M*N，要把它壓成 m*n: 我就把圖片分成(M*N)/(m*n)個長方塊，然後針對其值做(INTER_CUBIC 就是做立方插值)。

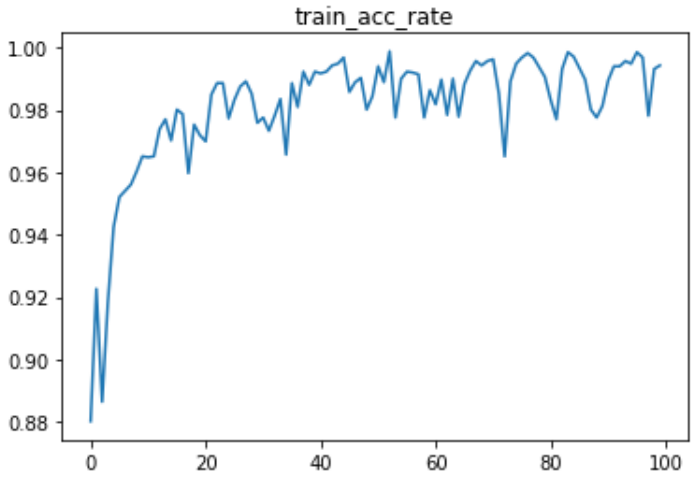
ii. 先做 convolution 以 kernel size = 3，把 input channel = 3 展成 output channel = 16，再做 ReLU、size 為 4*4 的 maxpooling，再將 16*7*7 的資訊用 fully connected 壓成 3 維的資訊。其中 w, b 參數在於 16*7*7 轉成 3 維資訊那裏。

使用 learning rate 固定時:

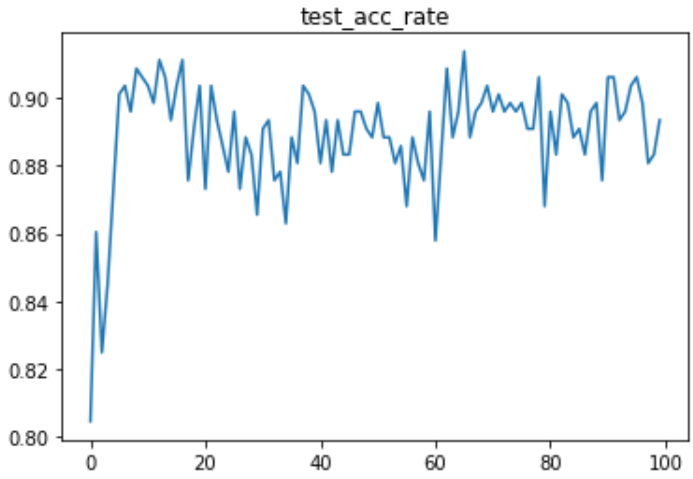
(1) Learning curve



(2) Training accuracy rate



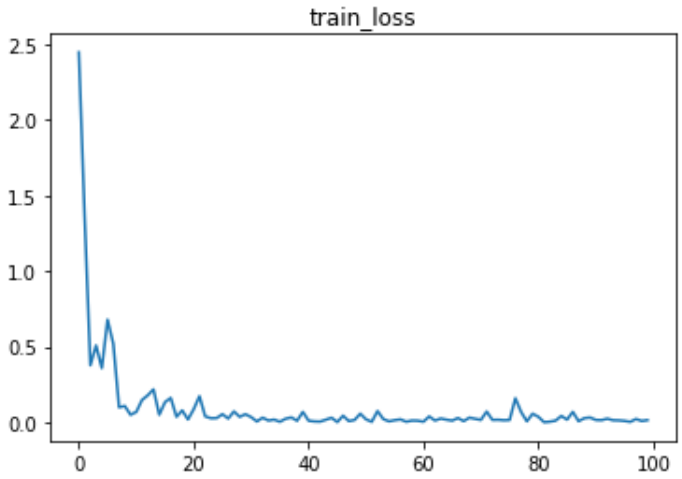
(3) Testing accuracy rate



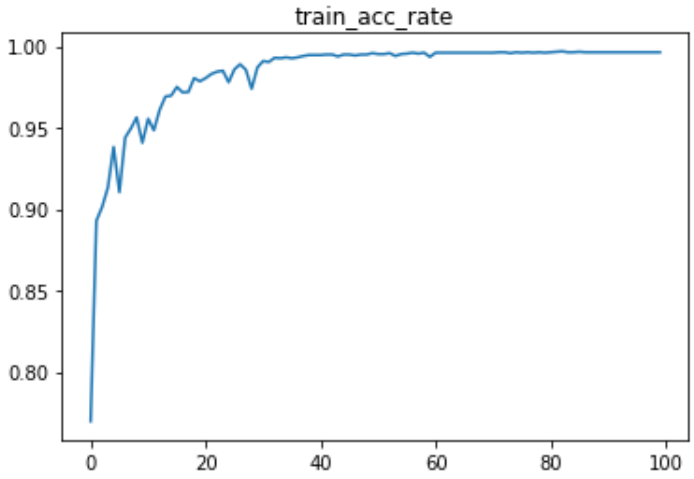
雖然 train 數值有有下降趨勢，但偶爾仍然會有上跳或下竄的問題。因此我將 learning rate 改低: 從原本 0.001 改成 0.0005(因為可能我本來的 learning rate 太大了)，並且使用: 動態 learning rate: 每 30 個 epoch 就降低 10 倍

看起來好多了:

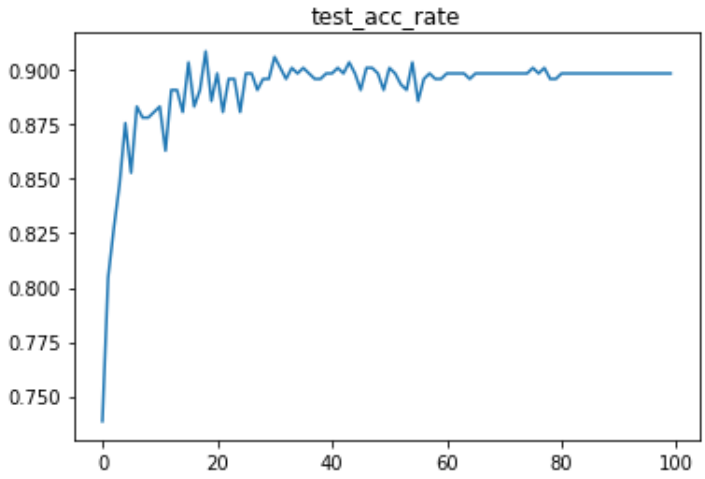
(1) Learning curve



(2)Training accuracy rate



(3) Testing accuracy rate



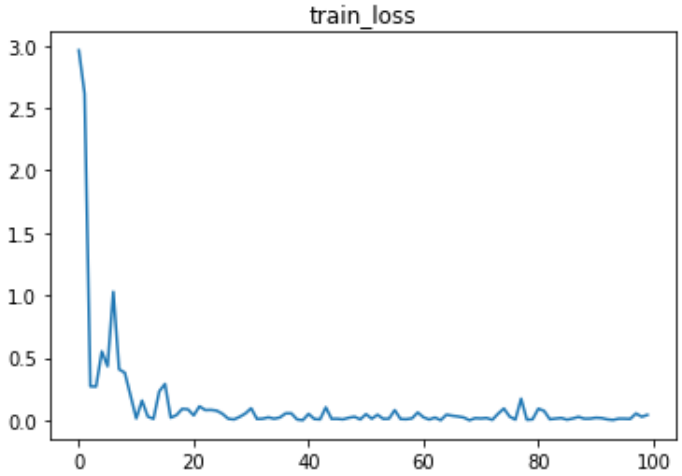
觀察 confusion matrix:

	0	1	2
0	74	6	9
1	4	8	10
2	5	6	272

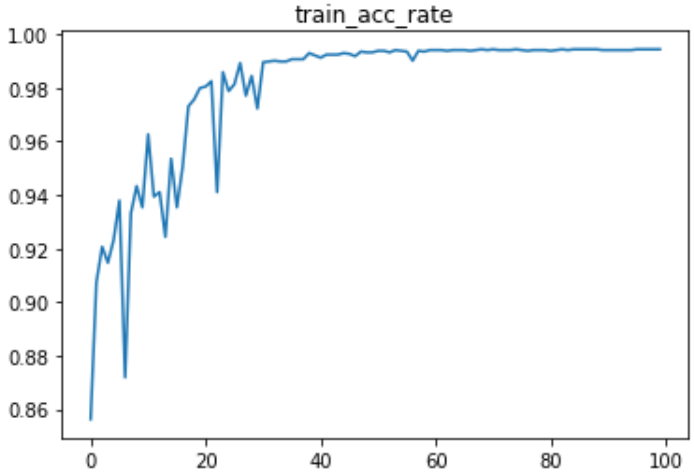
特別關注 none 那欄: 準確率只有 8/22 = 0.36 左右

iii. 以上一題的 confusion matrix 來看: none 那欄: 準確率只有 8/22 = 0.36 左右。原因就出在 imbalance data 那裏，就算 none 分錯了，在 loss function 裡面也是一小部分而已，影響 loss function 的大宗就是 good 那類。所以我在 loss function 上使用了加權 cross entropy: loss_func = nn.CrossEntropyLoss(weight=torch.FloatTensor([2,3,0.5]))。結果如下:

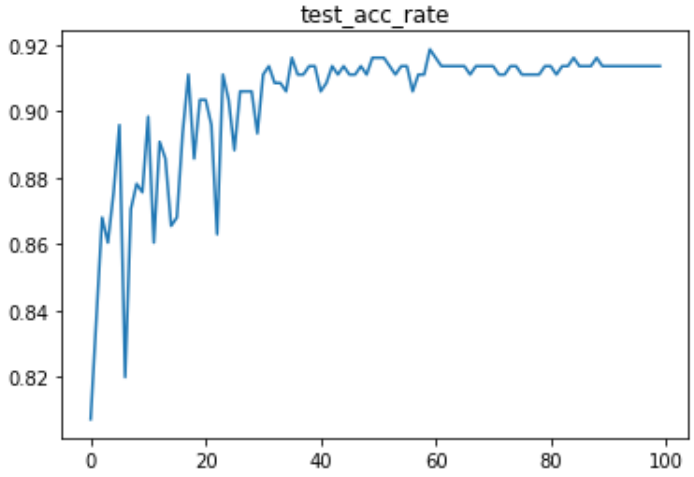
(1)Learning curve



(2)Training accuracy rate



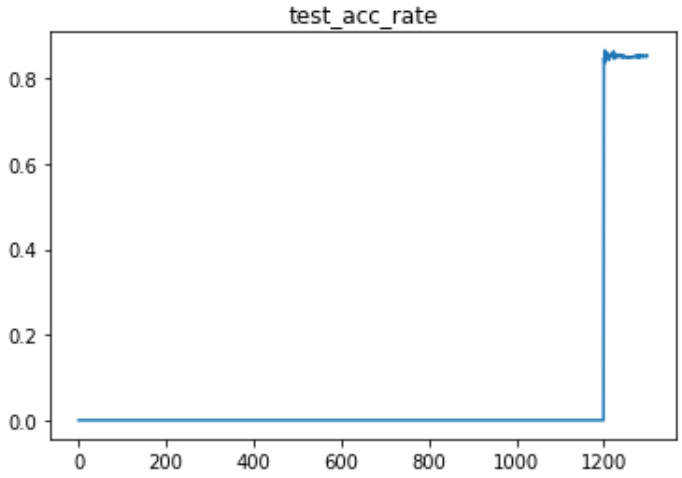
(3) Testing accuracy rate



觀察 confusion matrix:

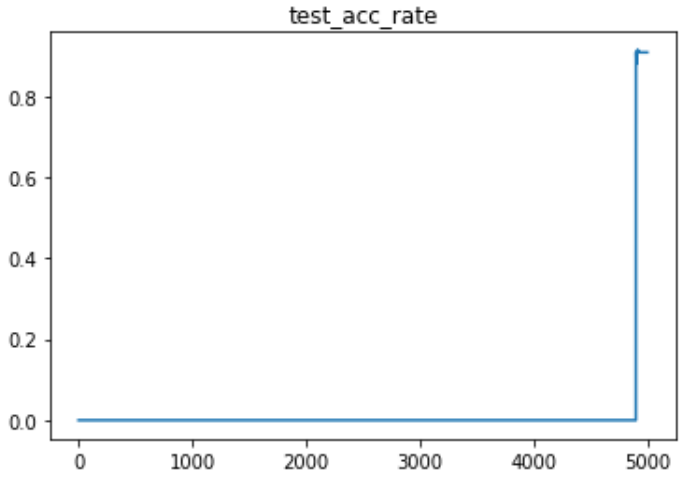
	0	1	2
0	78	3	8
1	7	10	5
2	7	4	272

不只整體正確率有上升，none 那欄: 準確率提升到 10/22 = 0.45 左右
但是加權這件事情就是多加了超參數，老實說我不覺得是一個穩健的方法。
因此我試了另外一個方法:resample
從 bad, none, good 各抽出 104 筆資料，總共抽出 312 筆資料，然後進行 training，重複抽 13 次。這樣讓模型在訓練的時候不會偏頗哪一個分類。同時我也重新抽 13 次，不要讓 good 中快 3000 筆資料只用到 104 筆。
這時看 train acc and loss 就沒甚麼意義了(因為每次都只訓練一個小部分的資料)
觀察 test accuracy rate and confusion matrix of test data:



	0	1	2
0	77	7	5
1	7	12	3
2	9	27	247

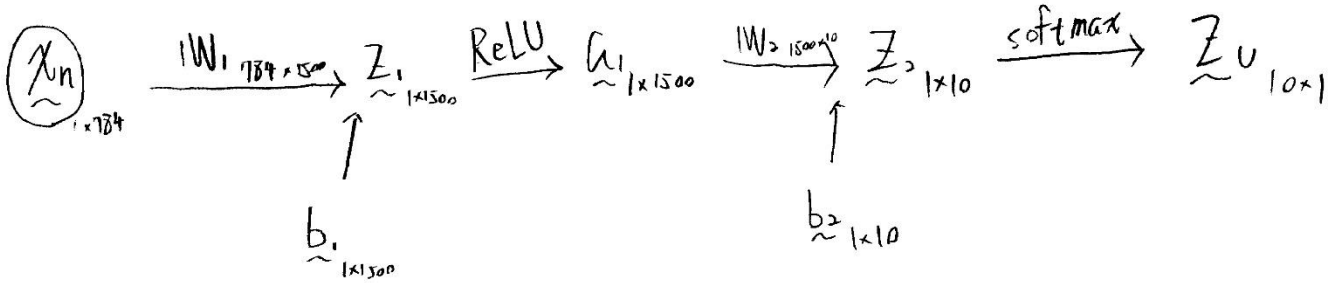
觀察 test data confusion matrix: 發現總體準確率反而下降，雖然 none 的準確率有提升，但相對而言，good 的準確率也下降許多。推測可能的原因為: 就算抽了 13 次，也沒辦法把所有 3000 筆 good 的資料都訓練到。
因此我想把 resample 的次數調高到 50 次並且加入動態 learning rate(每次 resample 會做 100 次 epoch, 根據這個 epoch 來做調整)
觀察 test accuracy rate and confusion matrix of test data:



	0	1	2
0	85	1	3
1	8	12	2
2	9	13	261

bad and good 的分類準確度有上升且 none 的準確率不變；雖然總體準確度有上九成，但與 resample 前的結果比較，仍然 good 的準確度仍然較低。

iv. 題外話: 這次作業我發現一個很有趣的問題: 因為計中帳號只能用 terminal，對我來說太痛苦了。我們 lab 只有一個人會在 terminal 上架 jupyter 或是那種我們可以滑鼠點來點去的介面。但是他太忙了，沒時間幫我弄。所以我就打算用我們 lab 的 i9 來跑就好了(後來事實證明，輸入 data 不要太大(例如 3*128*128 就太大)，cnn 不要過 50 層，i7-4xxx+16GB RAM 都還可以跑)，但我發現，用我們那台 i9 讀檔案的時候會出一個很有趣的 bug:
特別只有兩張圖片不能讀取:
kolkata-india-09th-dec-2018-people-wear-pollution-mask-during-an-awareness-campaign-for-right-to-breath-against-air-pollution-credit-saikat-paulpacific-pressalamy-live-news-R7H3AF.jpg
kolkata-india-09th-dec-2018-people-wear-pollution-mask-during-an-awareness-campaign-for-right-to-breath-against-air-pollution-credit-saikat-paulpacific-pressalamy-live-news-R7H3AP.jpg
雖然把檔名改短就可以了。不過我還是很好奇為什麼會出現這個 bug，想問助教們有沒有碰過這類問題。我分別試著使用 cv2.imread、Img.open 讀檔，前者會直接輸出 NoneType 的空檔案，後者會出 Error: 找不到該檔案。
另外如果可以的話，是不是能請助教發出一個教學檔，教我們如何從 terminal 架 jupyter?
謝謝助教!



$$\underset{\sim}{Z}_1 = \underset{\sim}{x}_n * |W_1 + \underset{\sim}{b}_1 \;,\quad [a_1]_i = \frac{1}{1+e^{-[\underset{\sim}{Z}_1]_i}}$$

$$\underset{\sim}{Z}_2 = \underset{\sim}{a}_1 * |W_2 + \underset{\sim}{b}_2$$

$$[\underset{\sim}{Z}_U]_i = \frac{\exp([\underset{\sim}{Z}_2]_j)}{\sum_{j=1}^{10} \exp([\underset{\sim}{Z}_2]_j)}$$

loss function : $E(\theta) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \ln [\underset{\sim}{Z}_U]_k$

$$E(\theta)_{(n)} = -\sum_{k=1}^{k} t_{nk} \ln [\underset{\sim}{Z}_U]_k$$

$$\frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{z}_v} = \left(\frac{-\underset{\sim}{t}_n}{\underset{\sim}{z}_v} \right)^T \quad 1 \times 10$$

$$\frac{\partial \underset{\sim}{z}_v}{\partial \underset{\sim}{z}_2} = \begin{pmatrix} \frac{\partial [\underset{\sim}{z}_v]_1}{\partial [\underset{\sim}{z}_2]_1} & \cdots & \frac{\partial [\underset{\sim}{z}_v]_1}{\partial [\underset{\sim}{z}_2]_{10}} \\ \vdots & & \\ \frac{\partial [\underset{\sim}{z}_v]_{10}}{\partial [\underset{\sim}{z}_2]_1} & \cdots & \frac{\partial [\underset{\sim}{z}_v]_{10}}{\partial [\underset{\sim}{z}_2]_{10}} \end{pmatrix} \quad 10 \times 10$$

$$\Rightarrow \left[\frac{\partial \underset{\sim}{z}_v}{\partial \underset{\sim}{z}_2} \right]_{ii} = [\underset{\sim}{z}_v]_i - ([\underset{\sim}{z}_v]_i)^2 \;, \quad i = 1, \ldots, 10$$

$$\left[\frac{\partial \underset{\sim}{z}_v}{\partial \underset{\sim}{z}_2} \right]_{ij} = -[\underset{\sim}{z}_v]_i * [\underset{\sim}{z}_v]_j \;, \quad i, j \in \{1, \ldots, 10\}, \; i \neq j$$

$$\frac{\partial \underset{\sim}{z}_2}{\partial \underset{\sim}{a}_1} = (W_2)^T$$

$$\left[\frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{z}_2} \right]_i = \left(t_{ni} [\underset{\sim}{z}_v]_i - t_{ni} + \sum_{k \neq i} t_{nk} [\underset{\sim}{z}_v]_i \right)$$

$$= \left([\underset{\sim}{z}_v]_i - t_{ni} \right)$$

$$\frac{\partial [\underset{\sim}{Z_2}]_i}{\partial W_2} = \frac{\partial (\underset{\sim}{a_1} * [W_2]_{:,i})}{\partial W_2}$$

$$= \begin{pmatrix} 0 & \cdots & \underset{\sim}{a_1}^t & \cdots & 0 \end{pmatrix}$$

↑ column i

we know:

$$\frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{Z_2}} * \frac{\partial \underset{\sim}{Z_2}}{\partial W_2} = \sum_{i=1}^{10} \left[\frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{Z_2}} \right]_i * \frac{\partial [\underset{\sim}{Z_2}]_i}{\partial W_2}$$

and $\frac{\partial [\underset{\sim}{Z_2}]_i}{\partial W_2}$ 只在 column i 非 0

$$\Rightarrow \frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{Z_2}} * \frac{\partial \underset{\sim}{Z_2}}{\partial W_2} = \left(\left[\frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{Z_2}} \right]_1 * \underset{\sim}{a_1}^t, \ldots, \left[\frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{Z_2}} \right]_{10} * \underset{\sim}{a_1}^t \right)$$

1500×10

$$= \underset{\sim}{a_1}^t * \frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{Z_2}}$$

$$\frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{z}_2} = \left(\underset{\sim}{z}_L - \underset{\sim}{t}_n \right)_{1\times 10}$$

$$\frac{\partial \underset{\sim}{z}_2}{\partial \underset{\sim}{a}_1} = (W_2)^t$$

$$\frac{\partial \underset{\sim}{a}_1}{\partial \underset{\sim}{z}_1} = (M_1, \cdots, M_{1500})$$

which $M_i = \begin{cases} 1 & , [\underset{\sim}{a}_1]_i > 0 \\ 0 & , [\underset{\sim}{a}_1]_i \le 0 \end{cases}$

$\frac{\partial \underset{\sim}{z}_1}{\partial W_1}$ 與前項 $\frac{\partial \underset{\sim}{z}_2}{\partial W_2}$ 同

$$\Rightarrow \frac{\partial \underset{\sim}{z}_1}{\partial W_1} = (\underset{\sim}{x}_n)^t * \left(\frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{z}_2} * \frac{\partial \underset{\sim}{z}_2}{\partial \underset{\sim}{a}_1} * \frac{\partial \underset{\sim}{a}_1}{\partial \underset{\sim}{z}_1} \right)$$

$$\frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{b}_2} = \frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{z}_2} * \begin{pmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{pmatrix}_{10\times 10}$$

$$\frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{b}_1} = \frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{z}_1} * \begin{pmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{pmatrix}_{1500\times 1500}$$

Summary:

$$\frac{\partial E(\theta)_{(n)}}{\partial W_2} = \underset{\sim}{a_1}^t * \left(\underset{\sim}{z_v} - t_n \right)^t$$

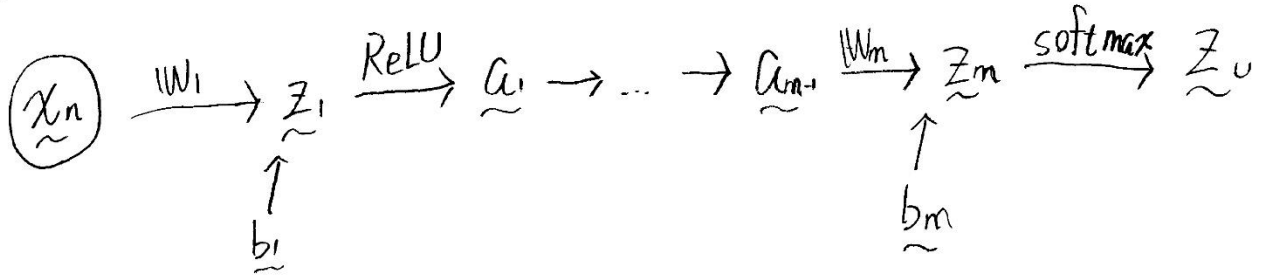
$$\frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{b_2}} = \left(\underset{\sim}{z_v} - t_n \right)^t$$

$$\frac{\partial E(\theta)_{(n)}}{\partial W_1} = (\underset{\sim}{x_n})^t * \left(\frac{1}{l_n^2} \underset{1\times 10}{\left(\underset{\sim}{z_v} - t_n \right)^t} * \underset{10\times 100}{(W_2)^t} * \frac{\partial \underset{\sim}{a_1}}{\partial \underset{\sim}{z_1}} \right)$$

$$\frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{b_1}} = \left(\left(\underset{\sim}{z_v} - t_n \right)^t * (W_2)^t * \frac{\partial \underset{\sim}{a_1}}{\partial \underset{\sim}{z_1}} \right)$$

In general:

若今天有



$$\frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{z}_m} = (\underset{\sim}{z}_v - t_n)^t$$

$$\frac{\partial E(\theta)_{(n)}}{\partial |W_m} = \underset{\sim}{a}_{m-1}^{\,t} * (\underset{\sim}{z}_v - t_n)^t$$

$$\frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{b}_m} = \frac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{z}_m}$$

for $k = 2, \dots, m$

$$\begin{cases} \dfrac{\partial \underset{\sim}{z}_k}{\partial \underset{\sim}{a}_{k-1}} = (|W_k)^T, \quad \dfrac{\partial \underset{\sim}{a}_{k-1}}{\partial \underset{\sim}{z}_{k-1}} = \sigma'(\underset{\sim}{a}_{k-1}) \\ \dfrac{\partial E(\theta)_{(n)}}{\partial |W_{k-1}} = \underset{\sim}{a}_{k-2}^{\,t} * \left(\dfrac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{z}_k} * \dfrac{\partial \underset{\sim}{z}_k}{\partial \underset{\sim}{a}_{k-1}} * \dfrac{\partial \underset{\sim}{a}_{k-1}}{\partial \underset{\sim}{z}_{k-1}} \right) \\ \dfrac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{b}_{k-1}} = \dfrac{\partial E(\theta)_{(n)}}{\partial \underset{\sim}{z}_k} * \dfrac{\partial \underset{\sim}{z}_k}{\partial \underset{\sim}{a}_{k-1}} * \dfrac{\partial \underset{\sim}{a}_{k-1}}{\partial \underset{\sim}{z}_{k-1}} \end{cases}$$

for $\underset{\sim}{a}_0 = \underset{\sim}{x}_n$