Deep Learning Homework 2

0510894 電機 4D 翁紹恩

一 Recurrent Neural Network for Classification

1 Processing data:

a 由於此題是要預測新增確診人數的上升及下降，而原資料為總確診數，因此先計算出兩日總確診數間之差，並用此數據當作接下來的使用數據。

```
for j in range(len(country_matrix[i])-1):
    diff[i][j] = country_matrix[i][j+1] - country_matrix[i][j]
```

b 將所有國家的數據依序讀出後利用下式先計算兩兩國家間的相關性(correlation)，程式實作則是直接使用 numpy 的 corrcoef，當 correlation > threshold 時，才將此國家的數據納入訓練數據，以免有趨勢差很多的國家造成訓練結果的偏差，而在此我將 threshold 設為 0.5。圖 1 為所有國家相關性示意圖，由於國家數太多分布太密不好觀察，因此圖 2 為擷取前幾個國家之示例圖。

$$\mathrm{Correlation}(X,Y) = \frac{\mathrm{Cov}(X,Y)}{\sqrt{\mathrm{Var}(X)}\sqrt{\mathrm{Var}(Y)}}.$$

```
for i in range(len(country_matrix)):
    for j in range(len(country_matrix)):
        coef_v = np.corrcoef(diff[i], diff[j])
        coef_m[i][j]=coef_v[0][1]
```

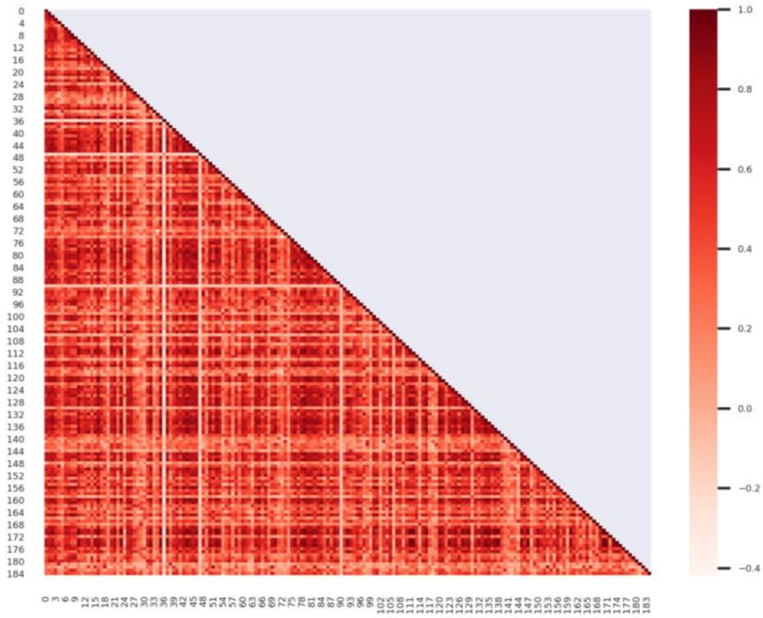


圖 1 所有國家間的 corelation 圖

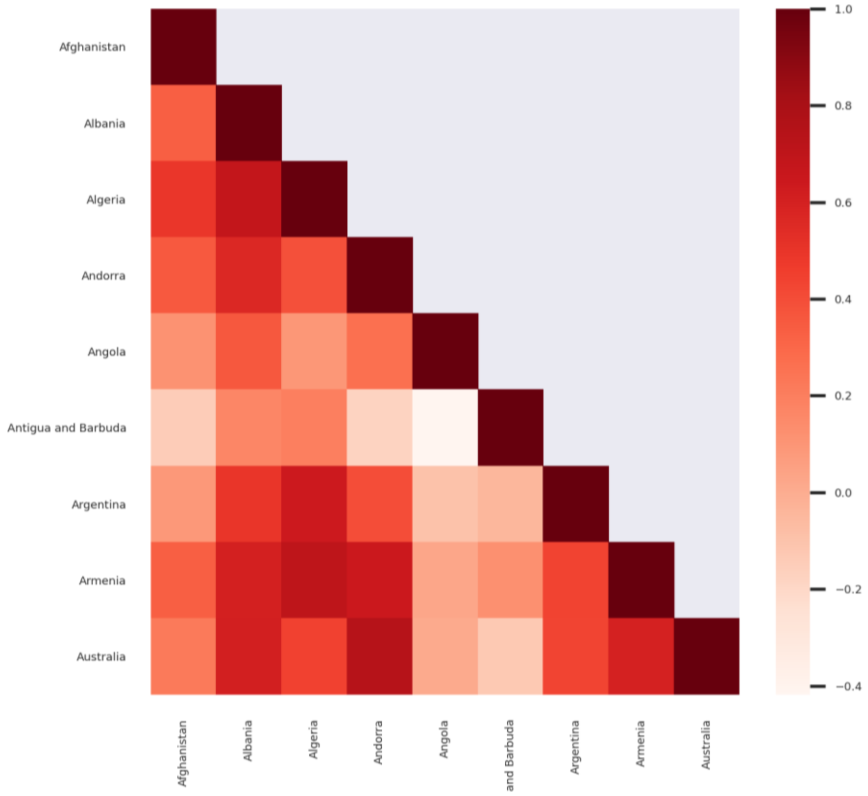


圖 2 擷取部分國家相關性示意圖

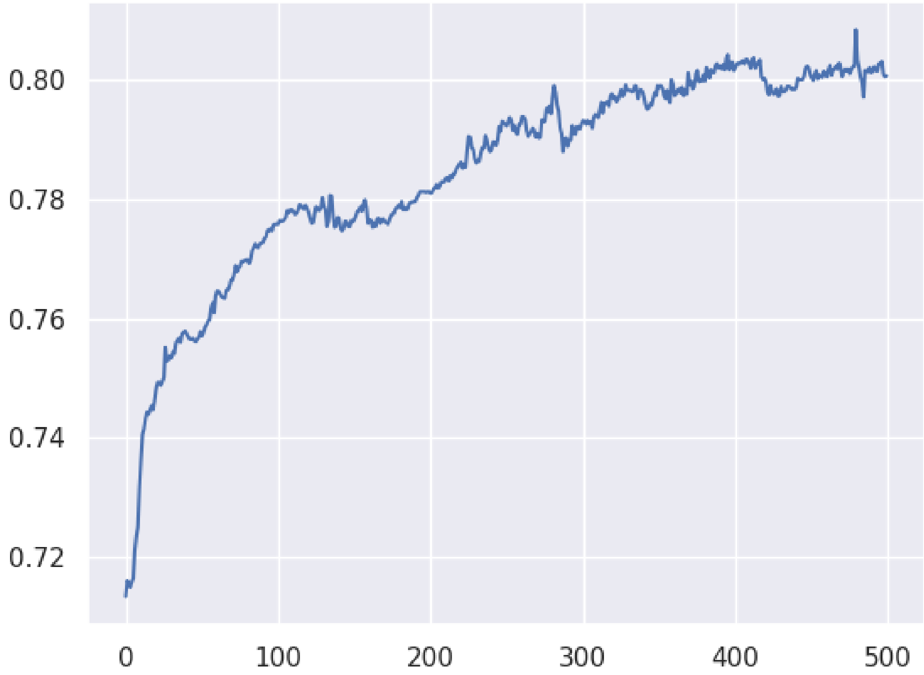
c 設定一段長度 L，為訓練數據所要提取出的長度，取數據的方法就像是用一個 window，依序往下一日移動，並對該 interval 最後一日做計算，若是新增人數仍大於前一日新增人數則標記 label 為 1，否則為 0，表示新增人數開始下降或是沒有改變。

d 最終數據集的一份輸入資料表示形式為如下：取 L 天為一個段落，且最後一天仍持續新增，則輸入 data 為 L 天的資料，出來結果 label=1。

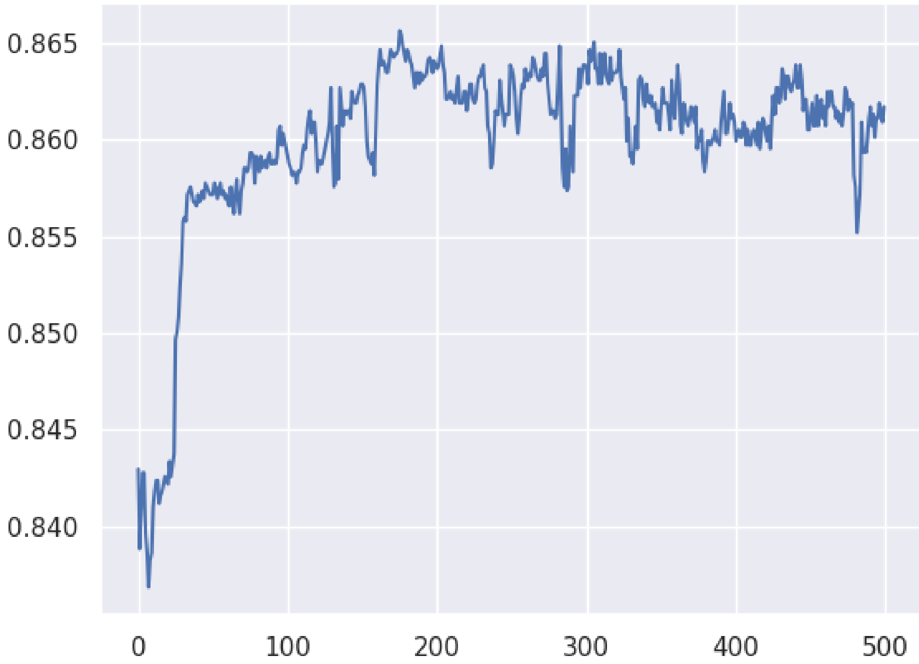
2 Build a recurrent neural network to predict the label based on the given sequence. And show the accuracy of training and test.

a 建立 RNN class Model，利用 torch.nn.RNN 並傳入 torch.nn.Linear 所建立的 fully connected layer 最後再通過一個 softmax，以分類是上升的機率或是下降的機率為多少。
RNN 的 input size 也就是特徵維度為 1，因為只有用一個數字表示；而 hidden_dim=12、n_layers=1，最後 fully connected layer 的 output_size=2，判斷是 0 或是 1 的機率，較高的為最終輸出。
而本次 batch size 直接使用整個 dataset。

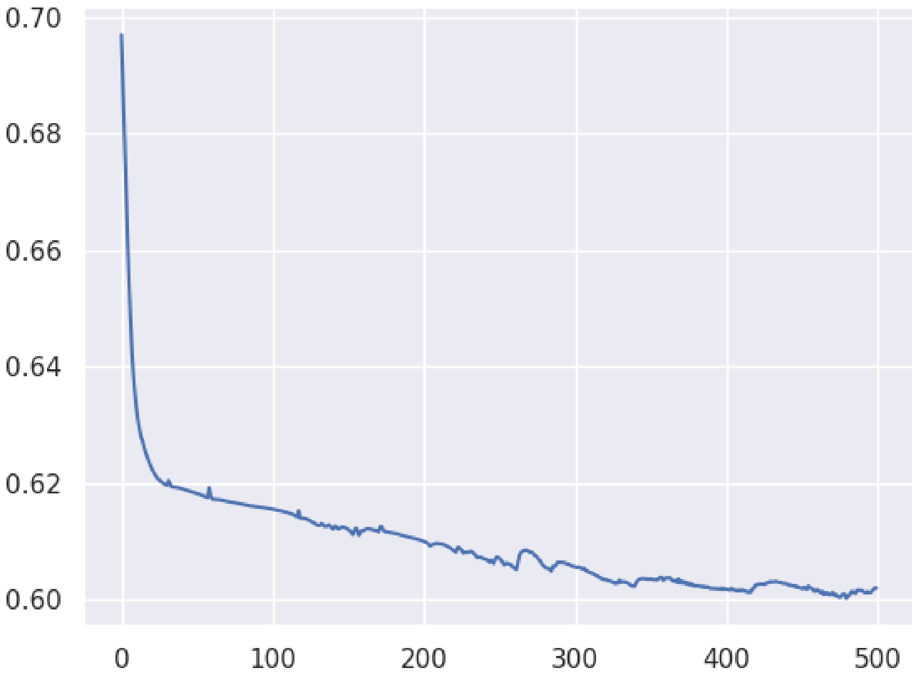
b Accuracy of training
橫坐標為 epoch 數



c Accuracy of testing



d Loss of train

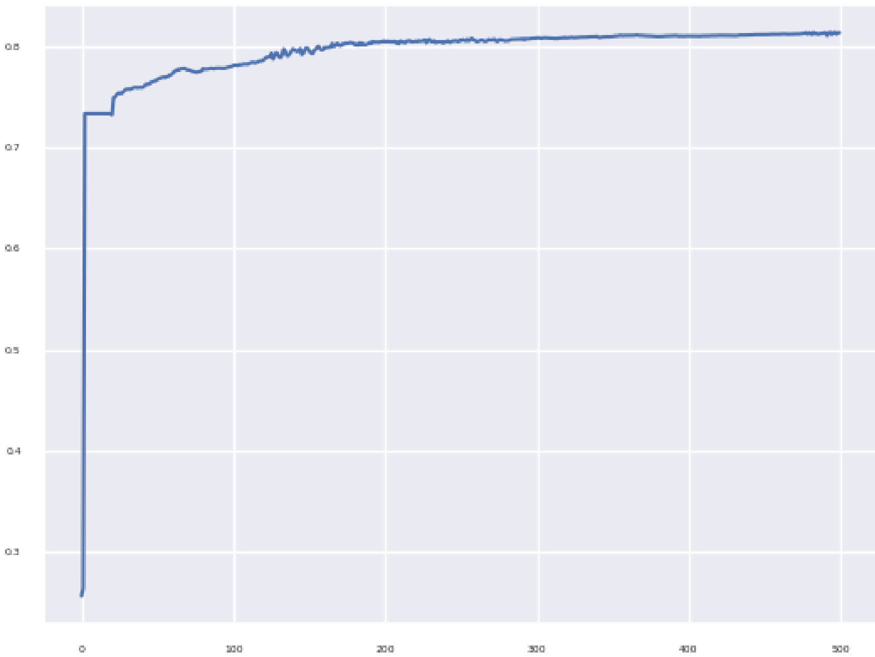


3 Please implement different recurrent neural network like LSTM and GRU and change the value of interval L to analyze its effect on the result

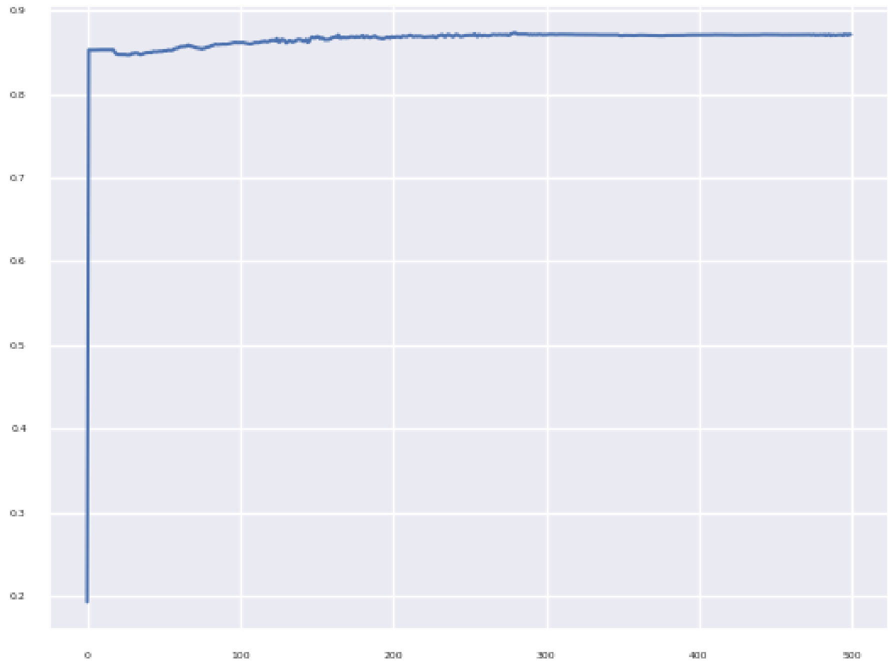
a 將原本 model torch.nn.RNN 換成 torch.nn.LSTM，參數設置大致上和 RNN 相同，除了 lstm 會在每一個神經元多一個開關 c。

b L=3

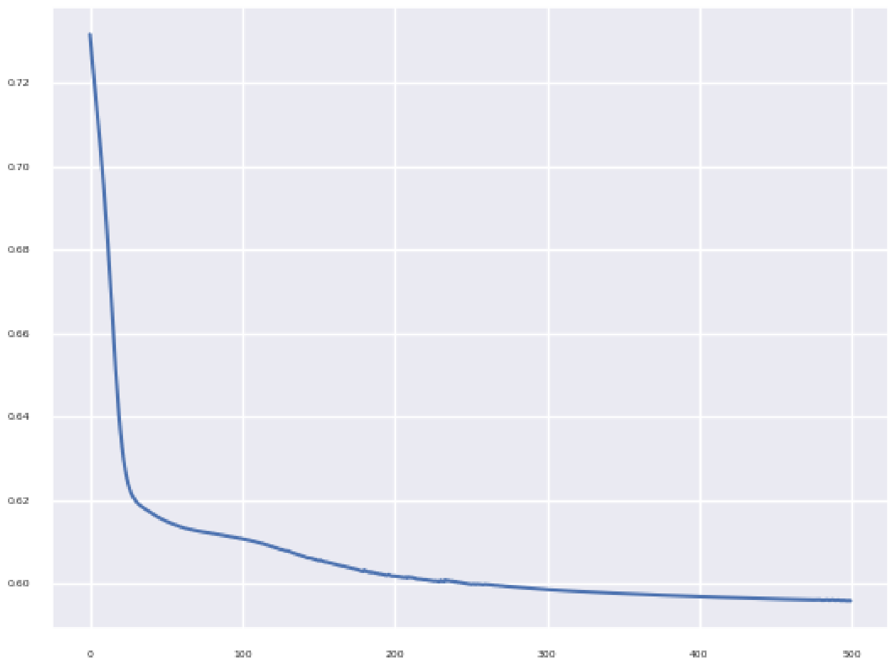
Training Accuracy



Testing Accuracy

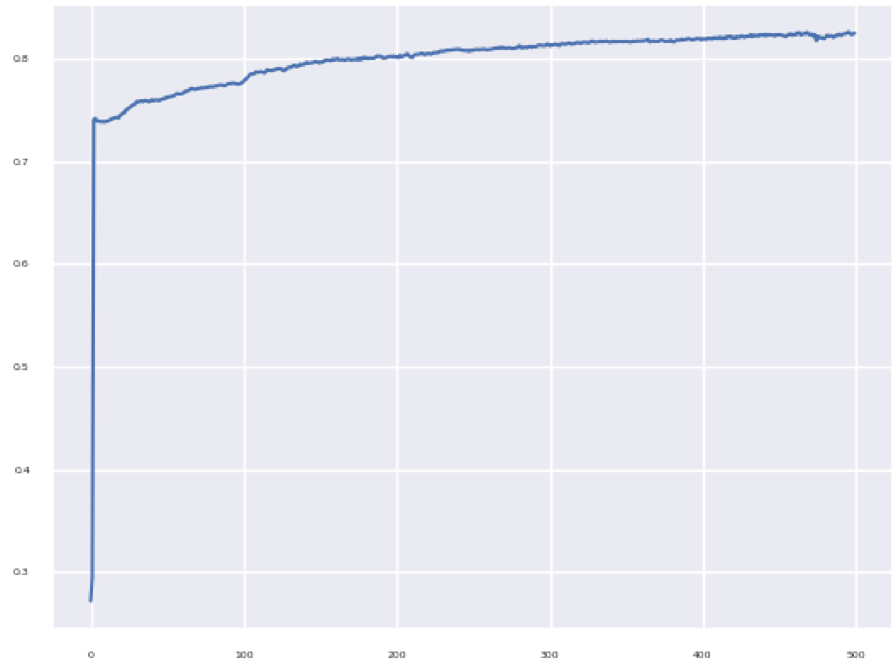

0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
100
200
300
400
500

Loss

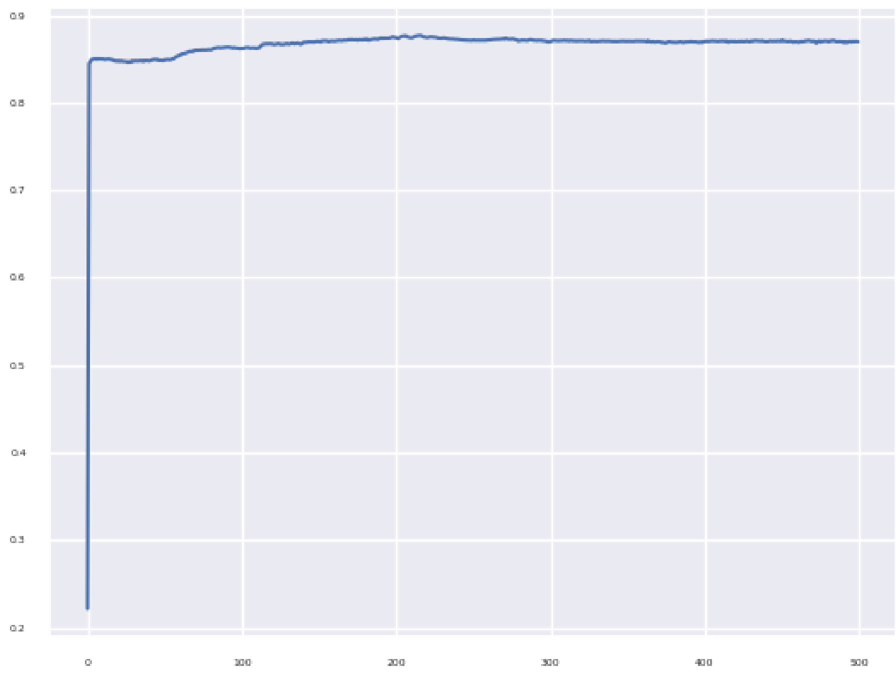

0.72
0.70
0.68
0.66
0.64
0.62
0.60
0
100
200
300
400
500

c L=5

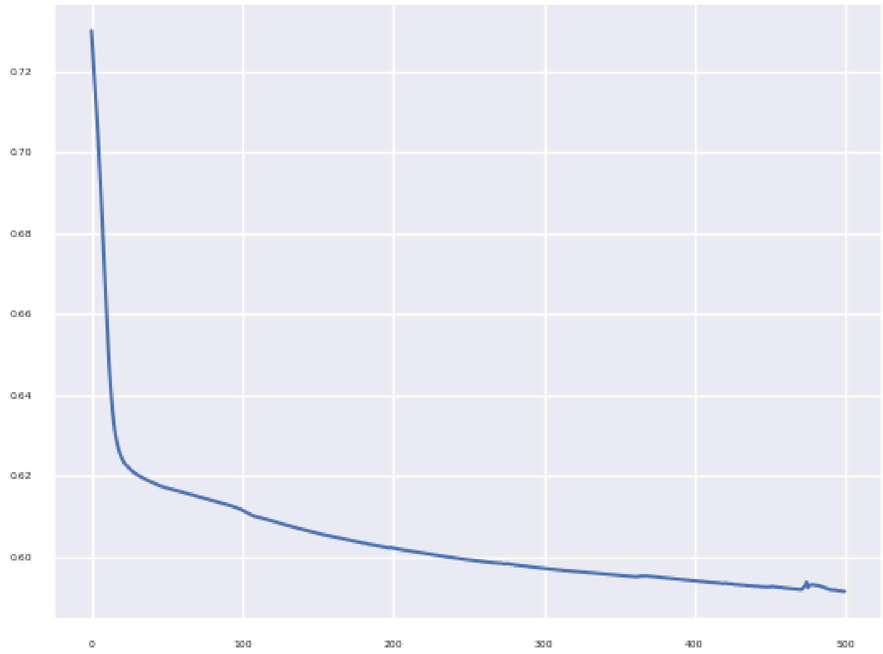
Training Accuracy



Testing Accuracy

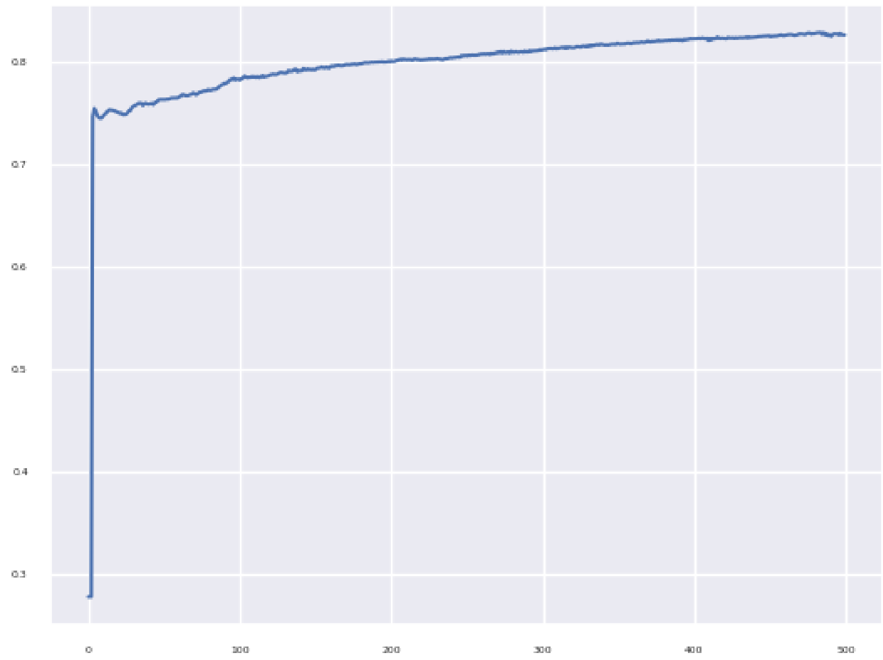


Loss

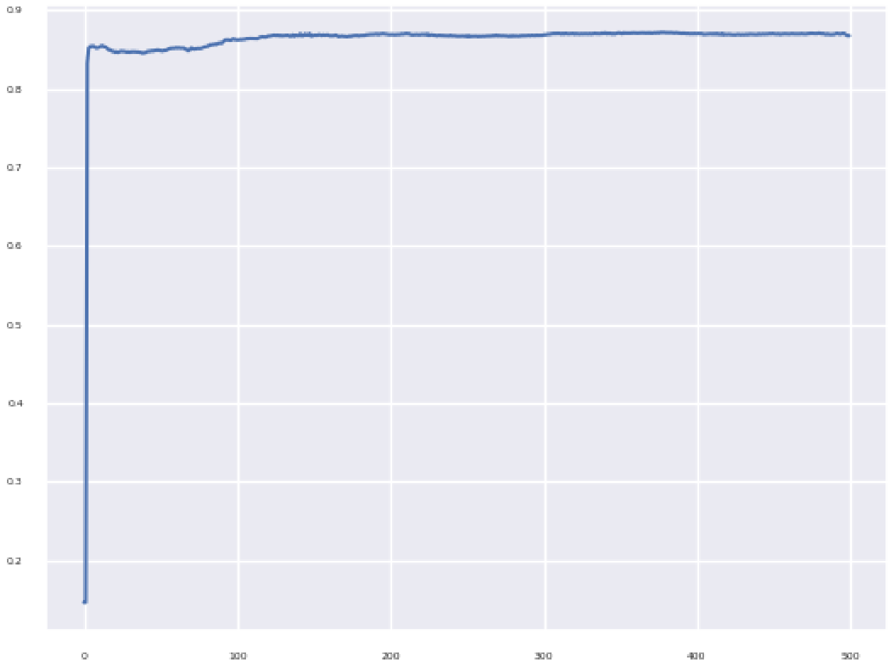


d L=7

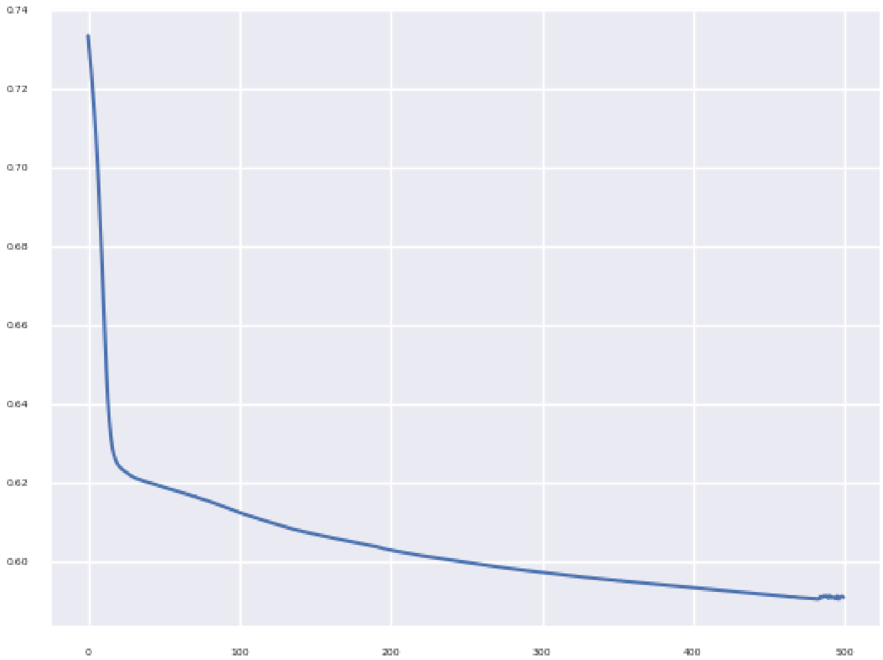
Training Accuracy



Testing Accuracy

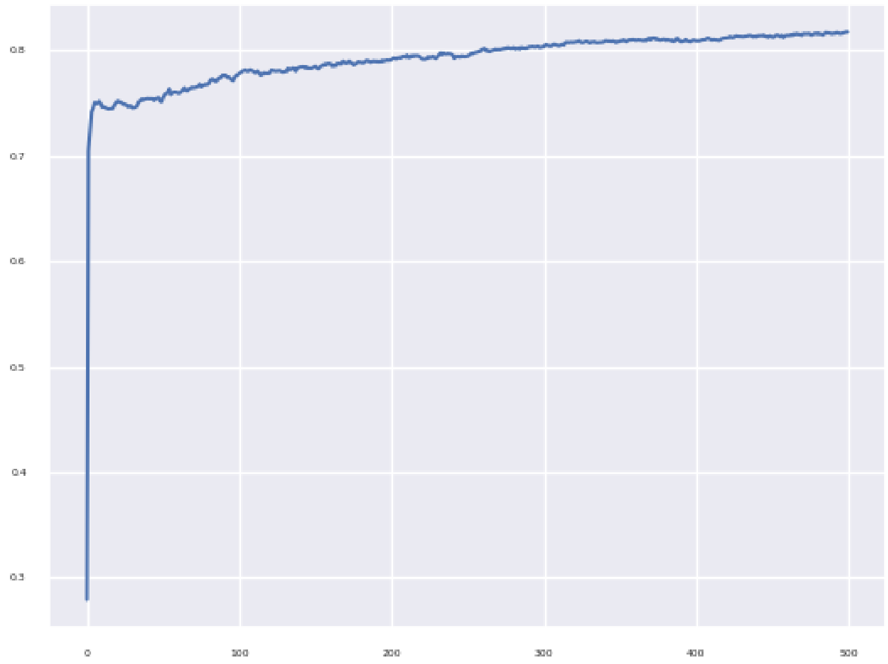


Loss

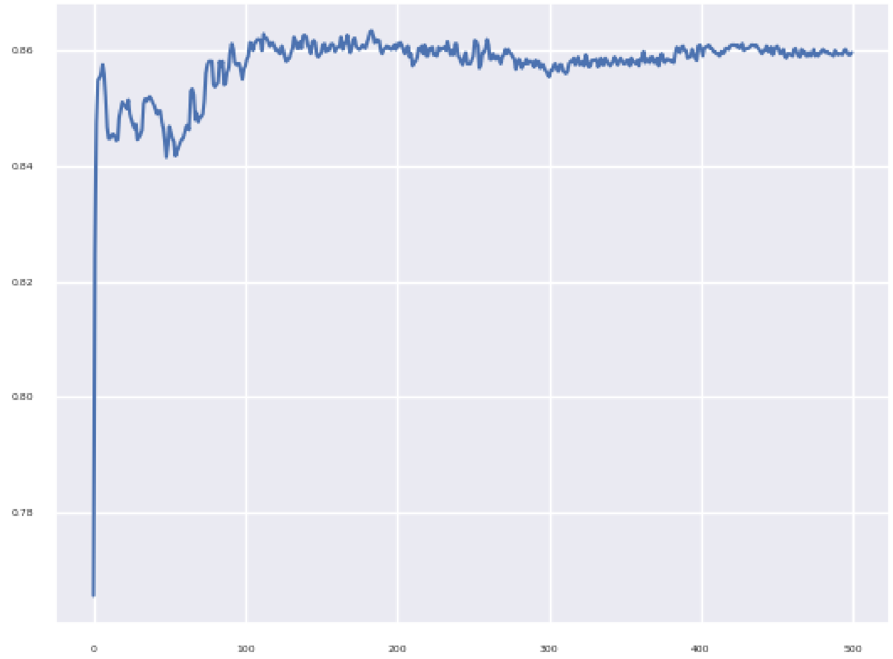


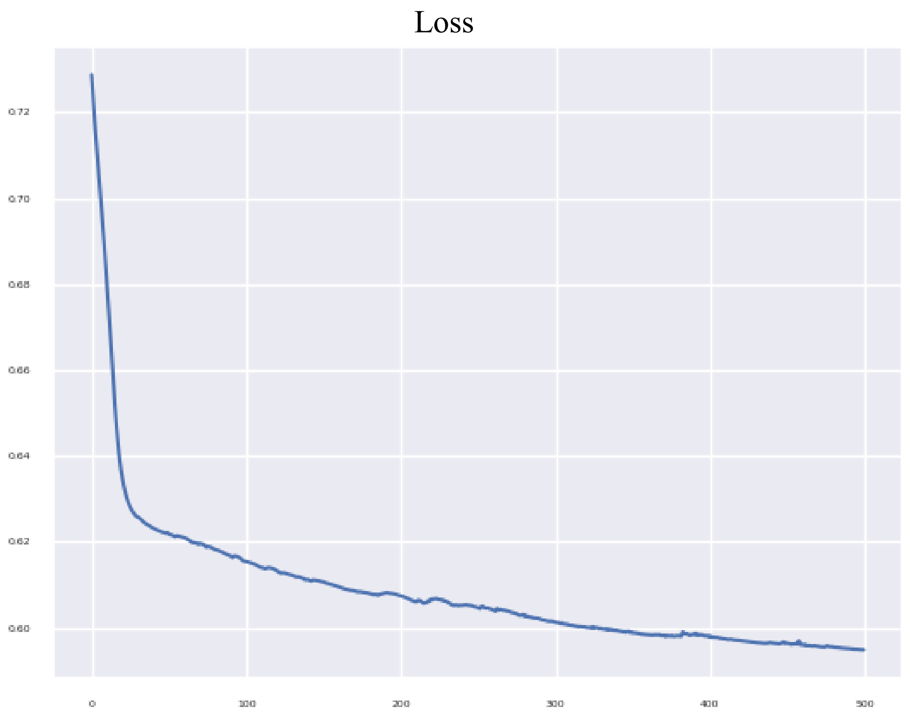
e L=9

Training Accuracy

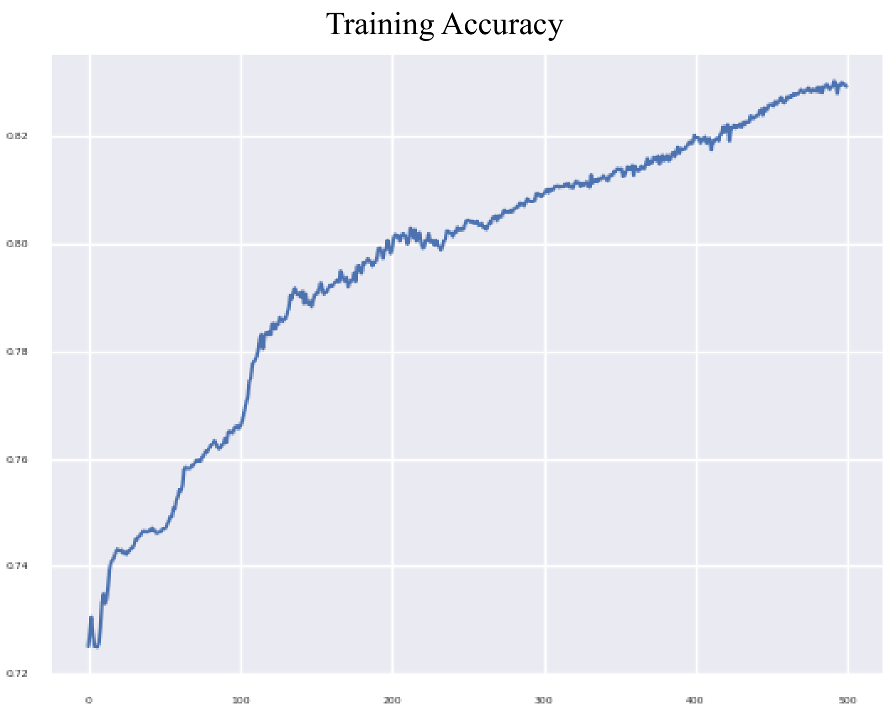


Testing Accuracy



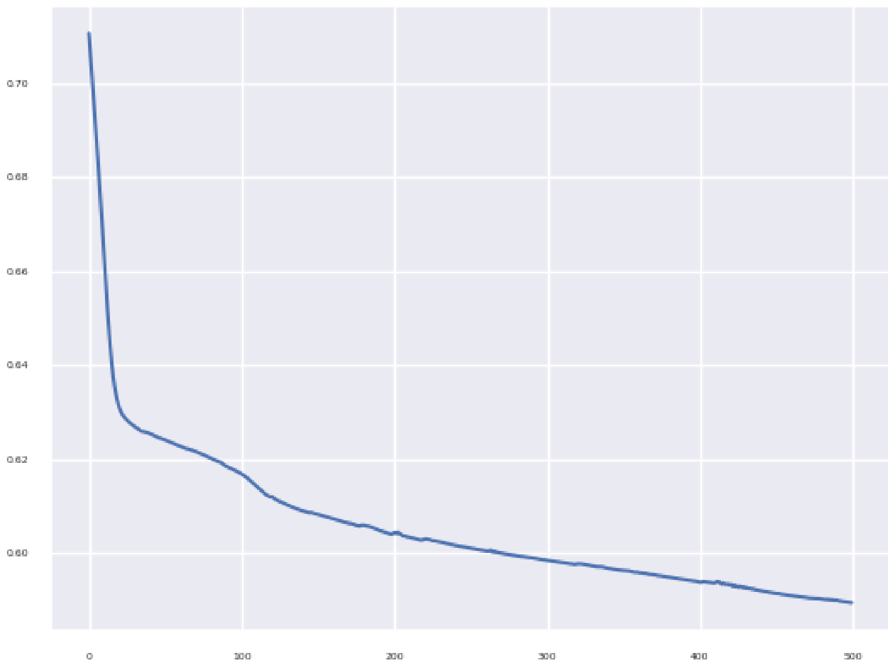

Loss
0.72
0.70
0.68
0.66
0.64
0.62
0.60
0
100
200
300
400
500

f L=10


Training Accuracy
0.82
0.80
0.78
0.76
0.74
0.72
0
100
200
300
400
500

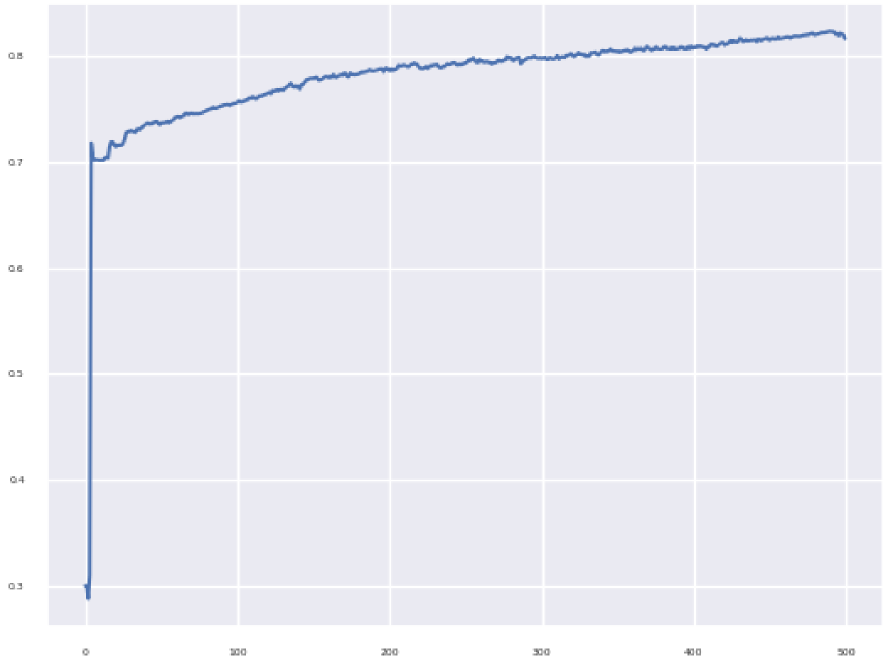
Testing Accuracy

Loss

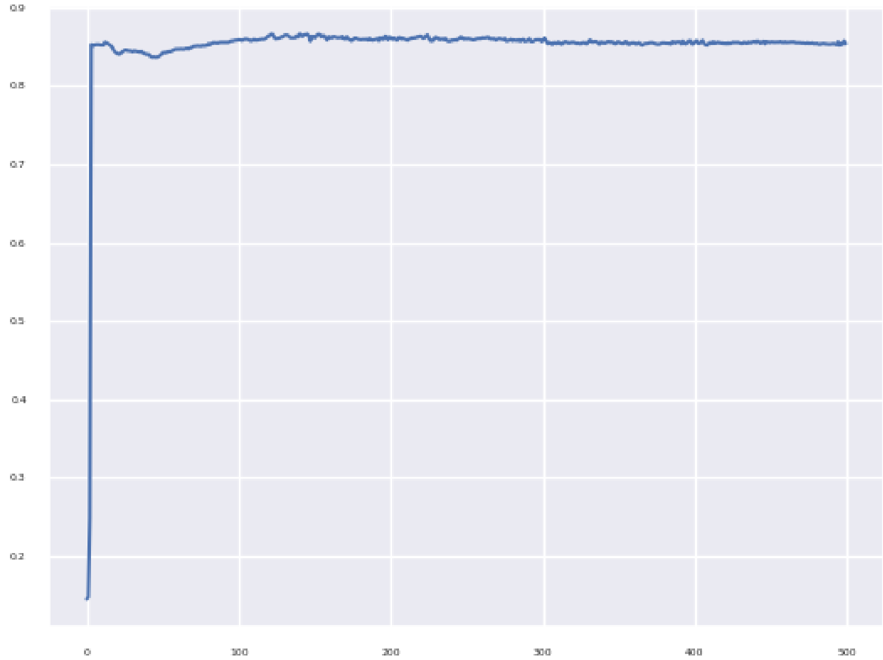


g L=15

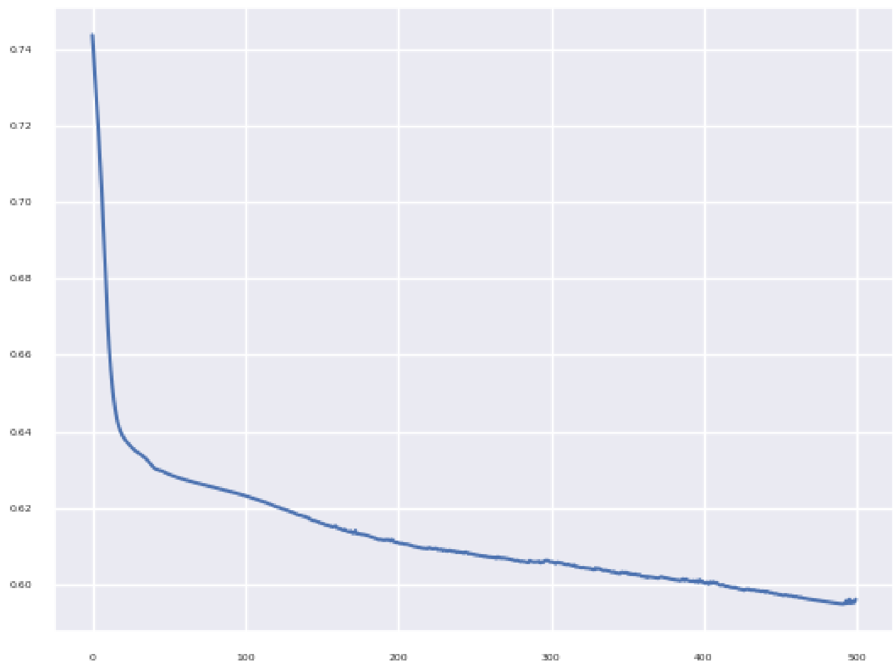
Training Accuracy



Testing Accuracy

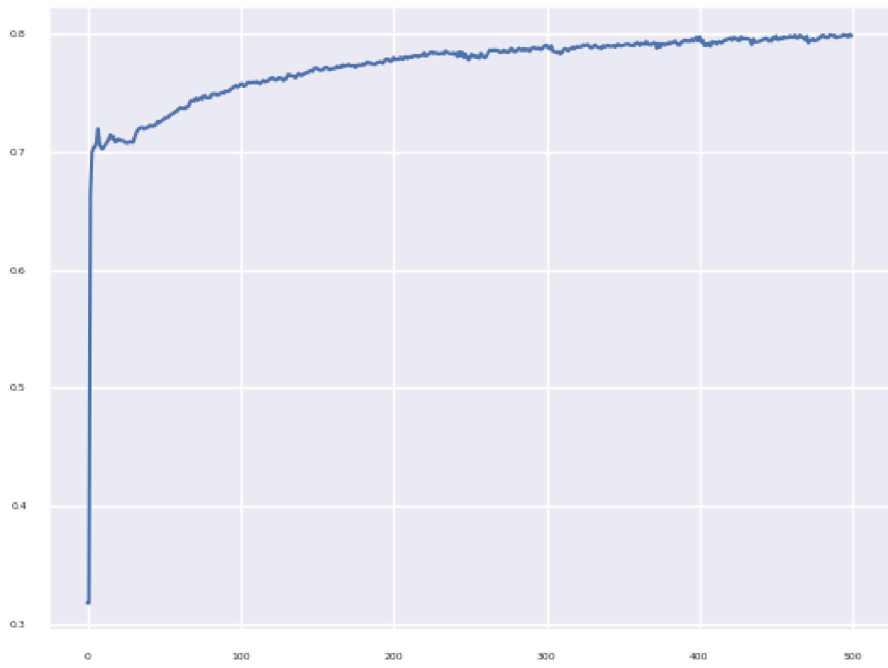


Loss

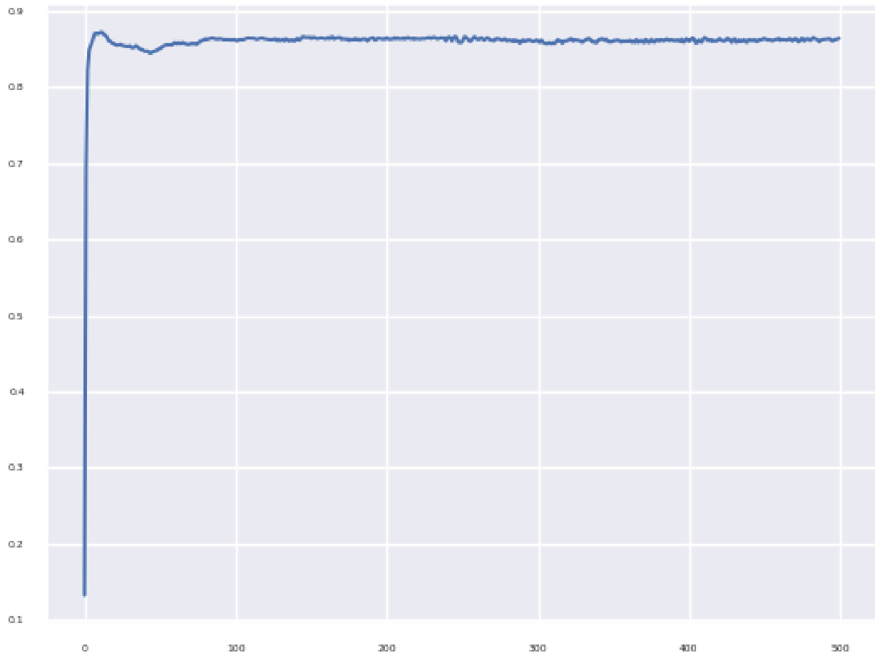


h L=20

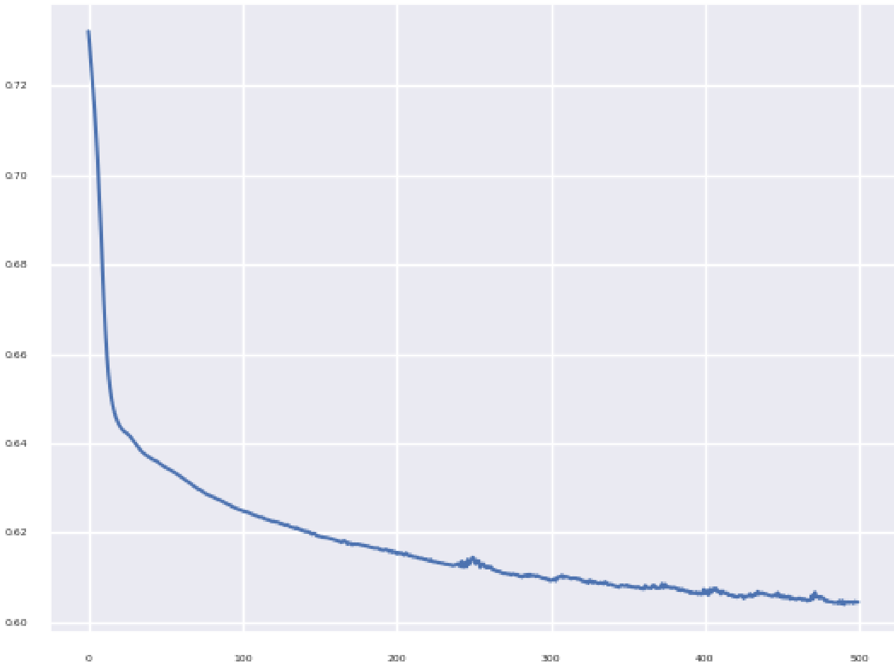
Training Accuracy



Testing Accuracy

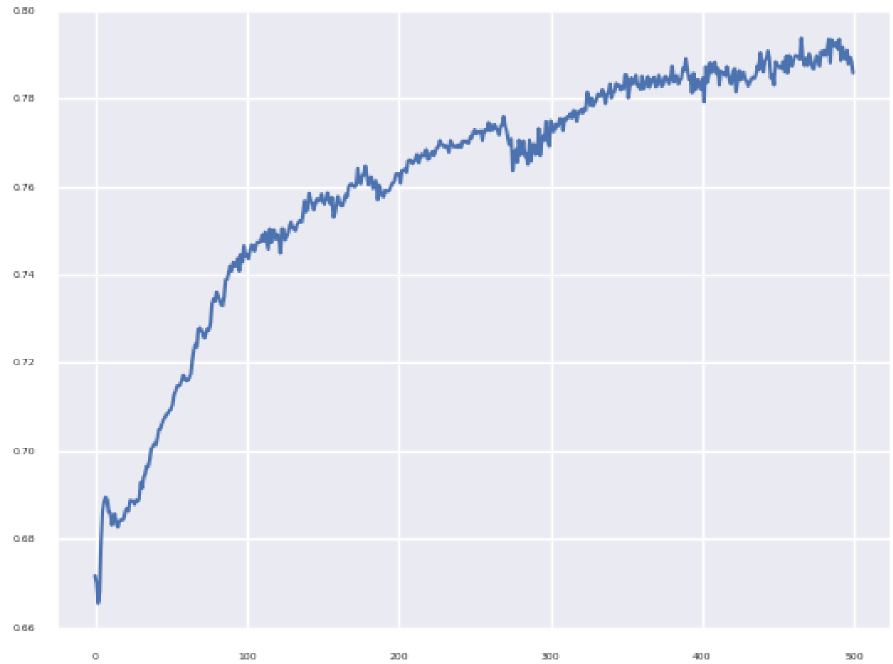


Loss



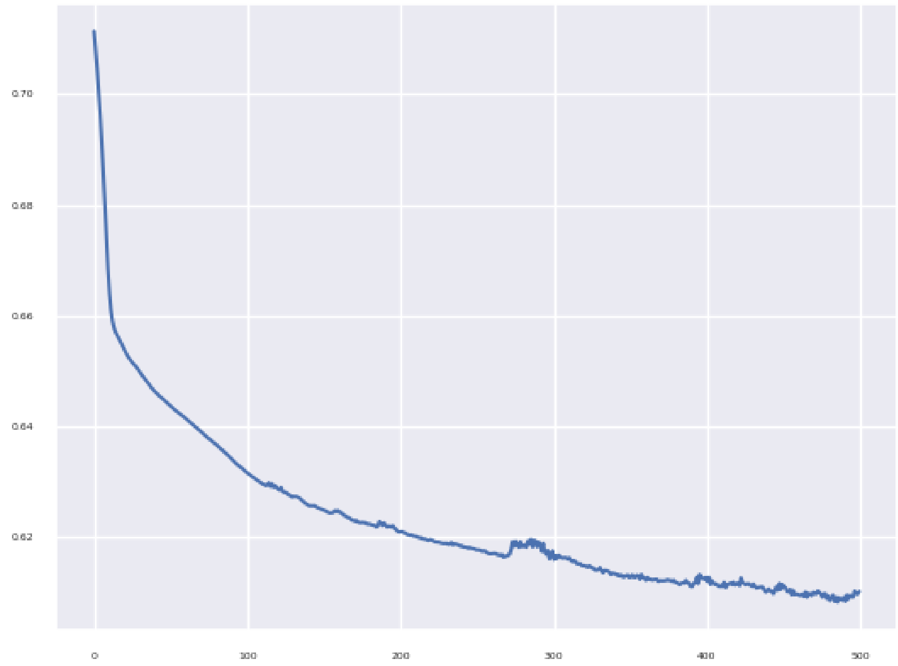
i L=30

Training Accuracy



Testing Accuracy

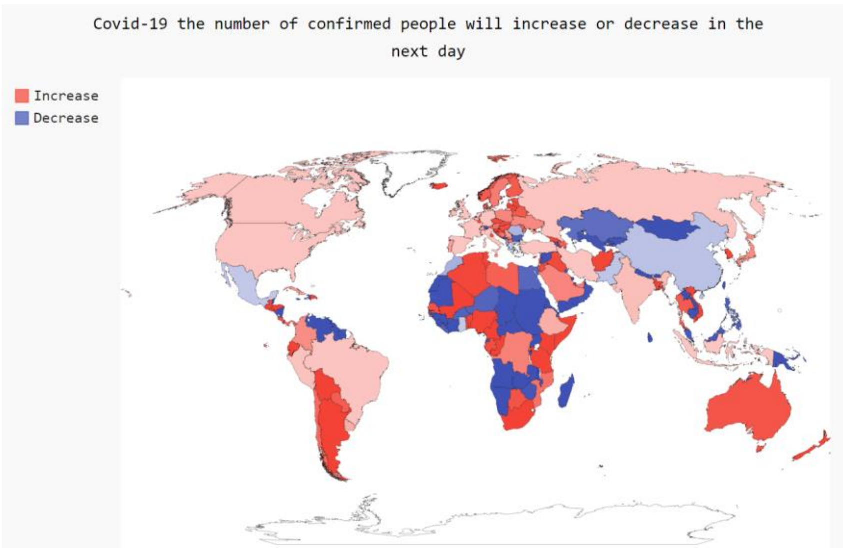
Loss



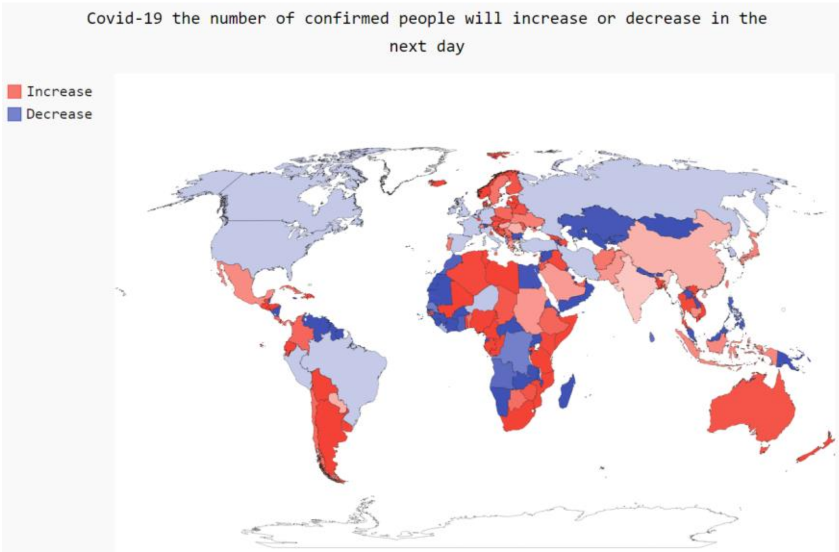
j　綜觀以上結果，可以發現 L 愈大，loss 曲線越平滑，但卻越有 overfitting 的現象，如 L=30，且在同樣的 epoch 數結束下 loss 最後數值較大。在最終的 test accuracy 除了 L=30 之外其餘並沒有差特別多大約都在 0.855~0.86 之間；L 較小時初始 accuracy 較低，L 大時初始 accuracy 偏高，導致 accuracy 看起來震盪幅度很大。

4　Compute the probability for each country and plot on a world map by using "pygal" package in python.

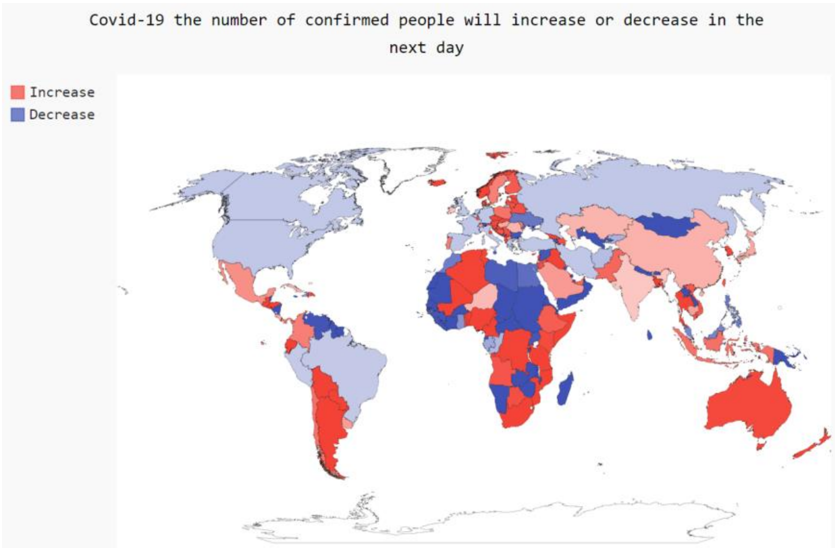
a　L = 3 (with LSTM)



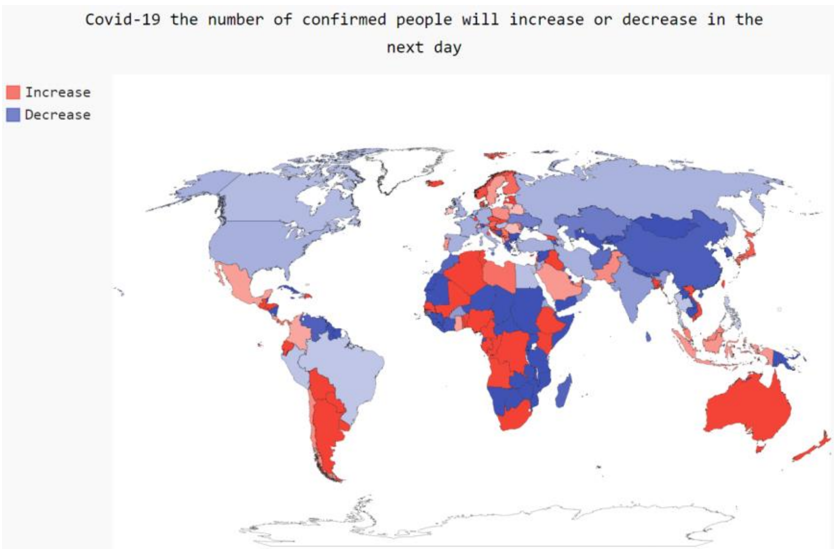
b　L = 5 (with LSTM)



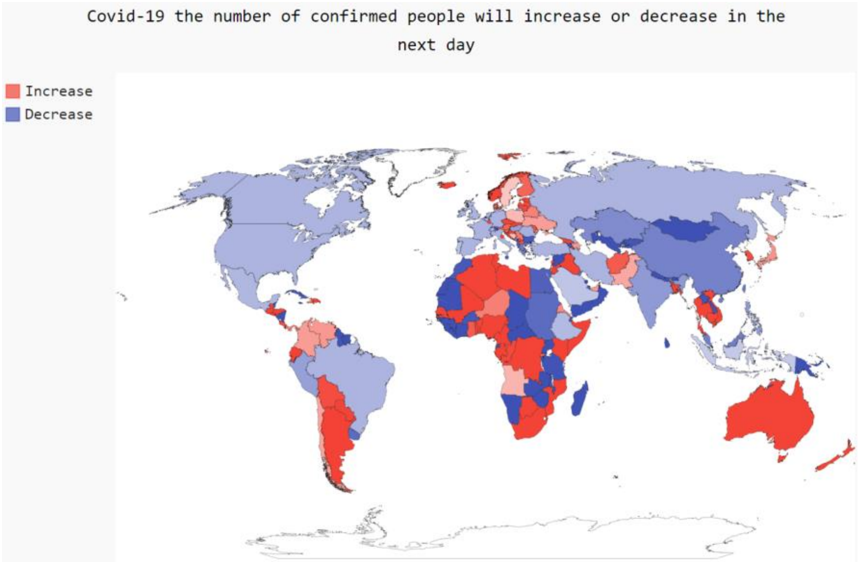
c L = 7 (with LSTM)



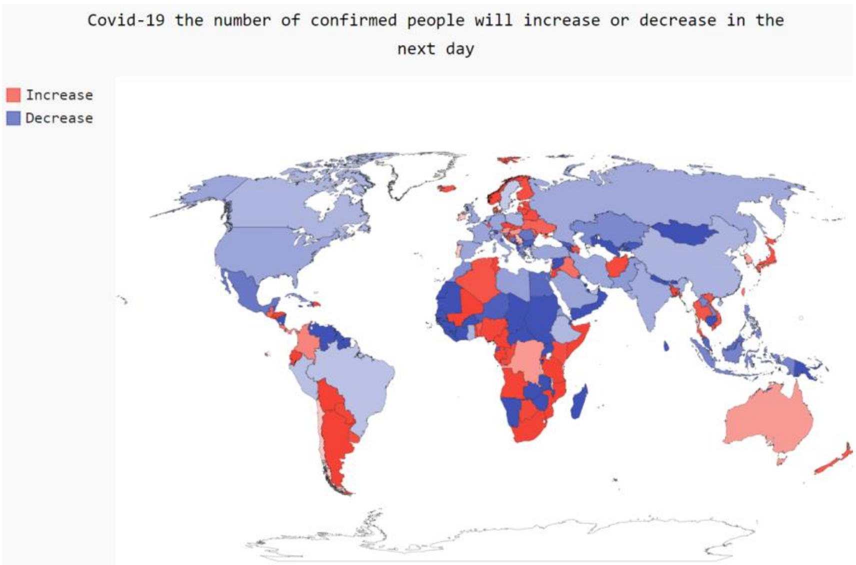
d L = 9 (with LSTM)



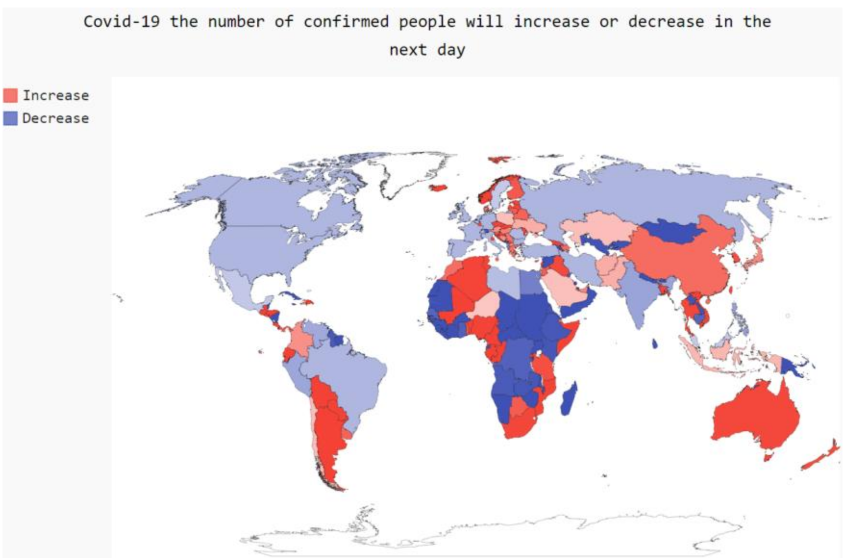
e L = 10 (with LSTM)



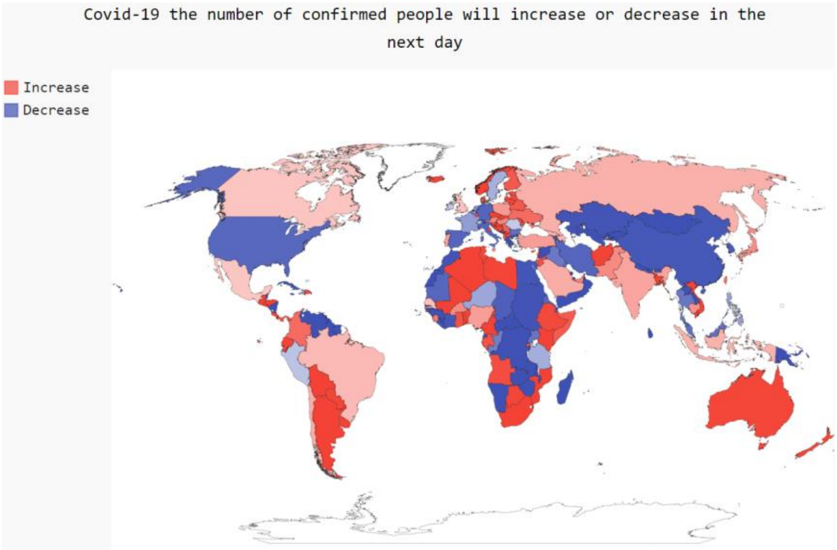
f L =15 (with LSTM)



g L = 20 (with LSTM)



h L = 30 (with LSTM)



5 Do some discussion based on your result.

以資料算出的 accuracy 大約在 L= 7 以前準確度較高，L 越大準確度反而下降，推測若是 L 值較小，數據只會根據附近幾天的數據因此受到其他的干擾因素較小，數據天數長可能會較有其他波動因素，造成準確度較為下降。

二 Variational Autoencoder for Image Generation

1 Describe in details how to preprocess images (such as resizing or cropping) and design the network architecture.

a Preprocess images

資料前處理僅將圖片由 64*64 變成 32*32，以減少訓練資料量；並將數據集以 7：3 分成 training 和 testing 兩種資料夾

b Network architecture

整個架構一個由 encoder 和 decoder 組成

i Encoder 組成：

① torch.nn.Linear

② torch.nn.ReLU

③ 將②產生出的結果分別通過一個 torch.nn.Linear 產生出平均值和取 log 的變異數

④ 通過平均值和取 log 的變異數產生出的高斯分布產生出要往下傳遞至 decoder 的數值

ii Decoder：接收 encoder 出來結果

① torch.nn.Linear

② torch.nn.ReLU

③ torch.nn.Linear

④ torch.nn.Sigmoid

⑤ 經過④之後的結果即為我們所要的輸出結果

iii Loss function：

① 使用 torch.nn.functional.binary_cross_entropy

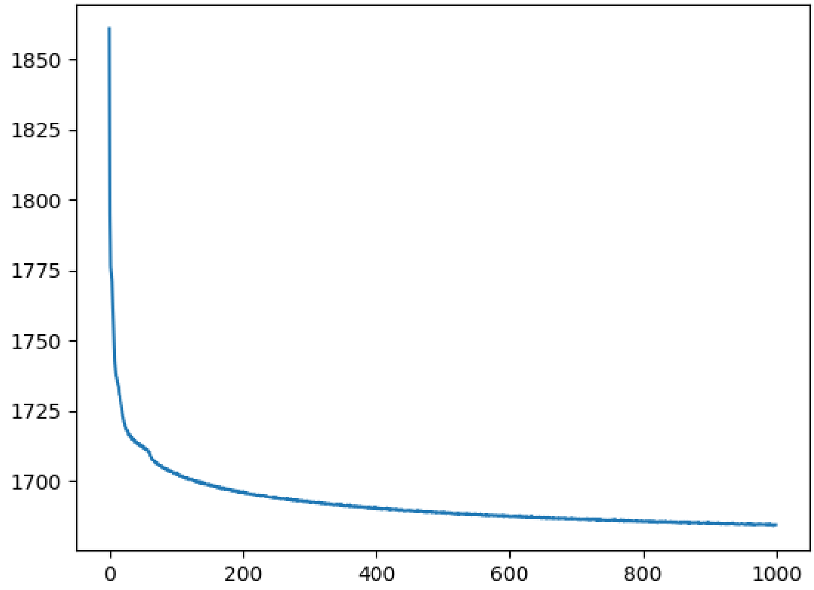
② 加上 KL diversion term

$$-\frac{1}{2}\sum_{j=1}^{J}\left(1+\log\sigma_j^2-\mu_j^2-\sigma_j^2\right)$$

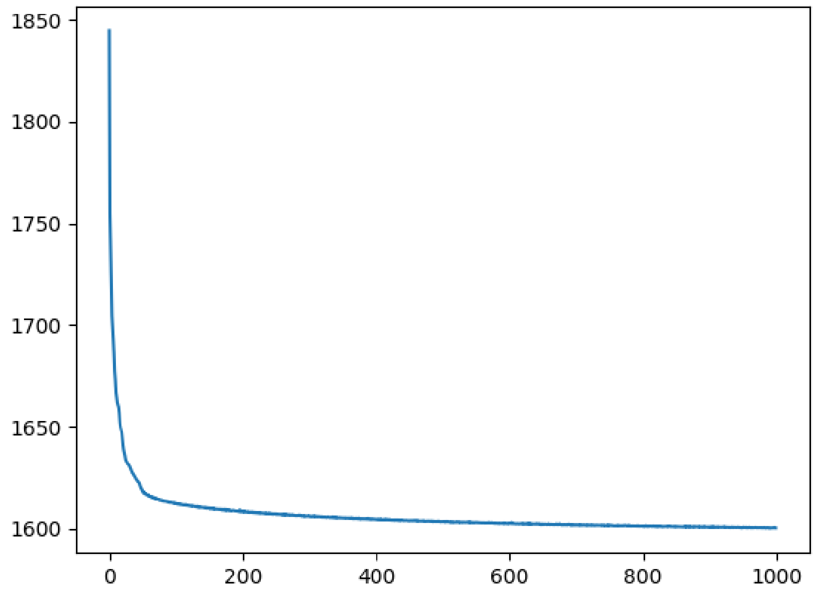
```
KLD = -0.5 * torch.sum(1 + logvar - mu.pow(2) - logvar.exp())
```

2 Plot the learning curve of the negative evidence lower bound (ELBO) of log likelihood of training images.

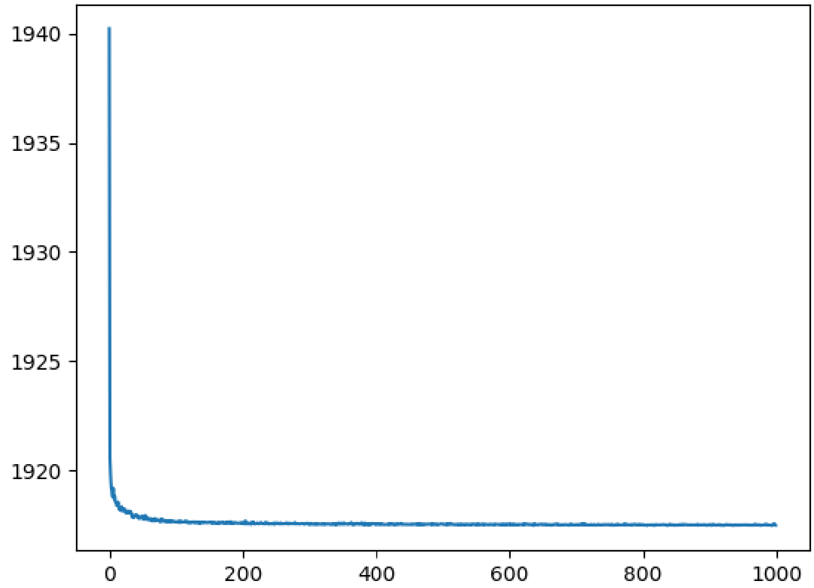
KL*1


1850
1825
1800
1775
1750
1725
1700
0
200
400
600
800
1000

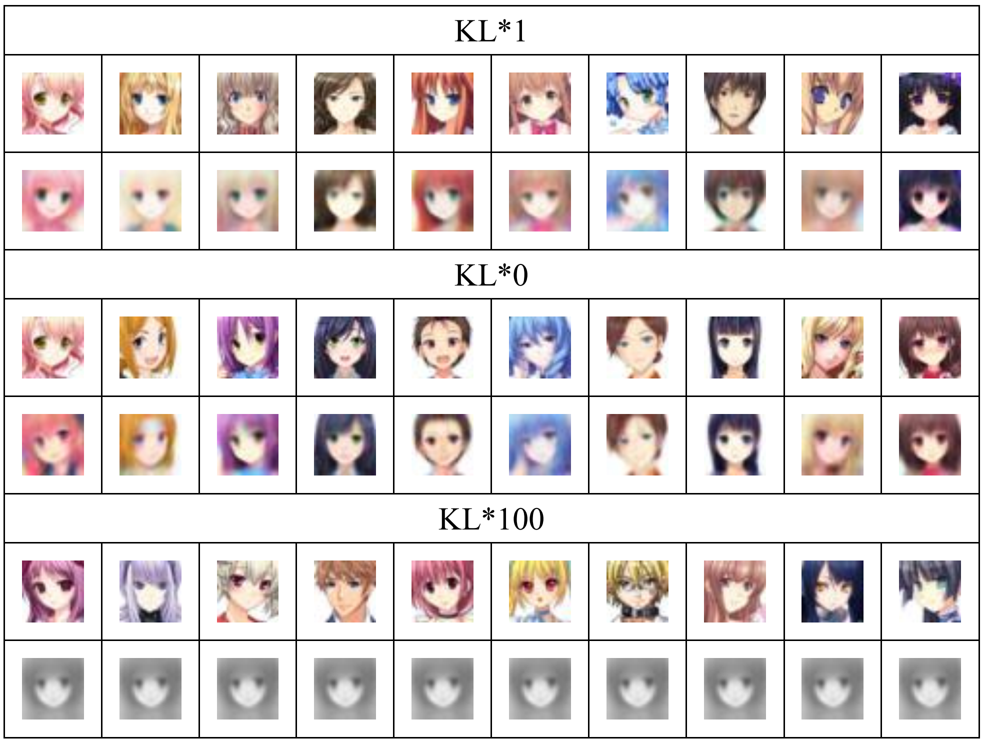
KL*0


1850
1800
1750
1700
1650
1600
0
200
400
600
800
1000

KL*100



3 Show some examples reconstructed by your model.(第一列為原圖)



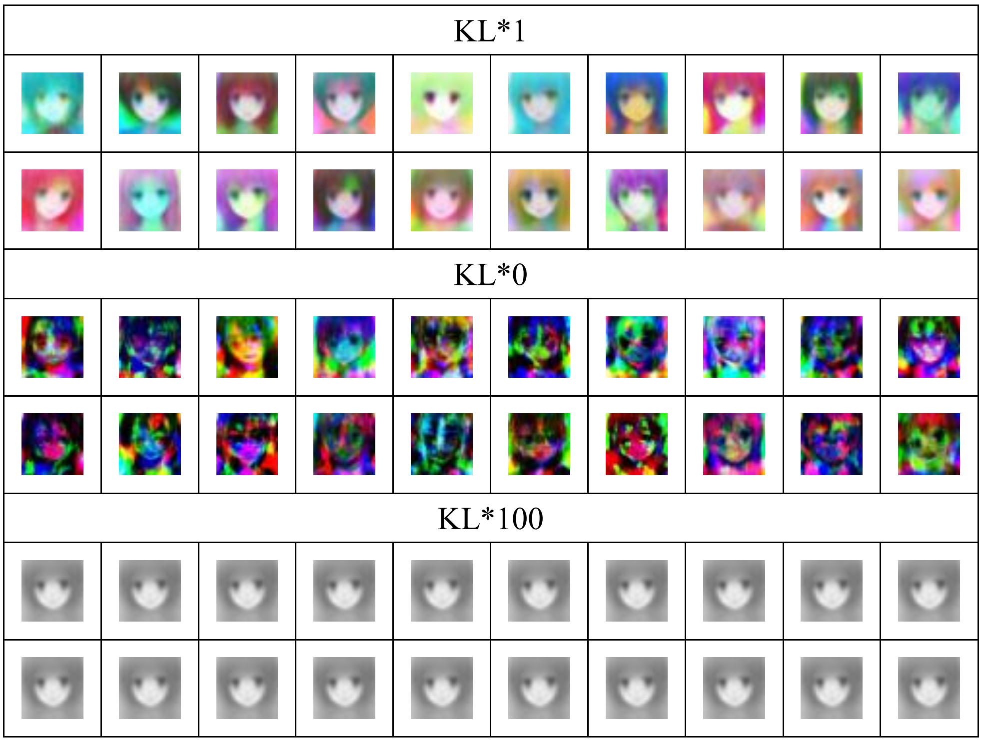
4 Sample the prior p(z) and use the latent codes z to synthesize some examples when your model is well-trained.(均取同樣 20 張圖)

a 由得出的 mean、variance 重新 random 出一組 prior

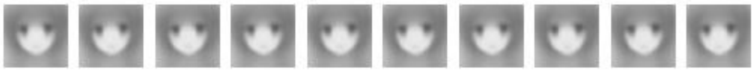
```
p_x = self.decoder(torch.normal(mu, torch.exp(0.5*logvar)))
```

b random 一組新的均值為 0 方差為 1 的常態分佈

```
p_x = self.decoder(torch.randn(3, 20))
```



5 Show the synthesized images based on the interpolation of two latent codes z between two real samples.

KL*1		
		
KL*0		
		
KL*100		
		

6 Do some discussion on the effect of KL term based on your result.

由以上結果可以觀察發現，當 KL*0 時，圖片還原的細緻度比 KL*1 時還要好，而 KL*100 時，圖片都只能產生出同一張人臉，儘管 loss 看似趨於收斂仍無法訓練成功；但是在 random 輸入 decoder 之後(4.b) KL*1 儘管在顏色上有偏差卻還可以還原出人臉顏色正確分布，KL*0 能還原出人臉特徵，但除了和原本顏色相差甚遠連人臉顏色分布區塊也十分混亂，而 KL*100 則完全無法還原，只學到同一張灰色的臉。因此就整體結果論，KL 不可或缺卻也不可占太大比重，我認為 KL 的比重參數也是一個可以學習的參數。