國立台灣科技大學電子工程系

基於深度學習之影像辨識

期末報告

人臉生成 with DCGAN

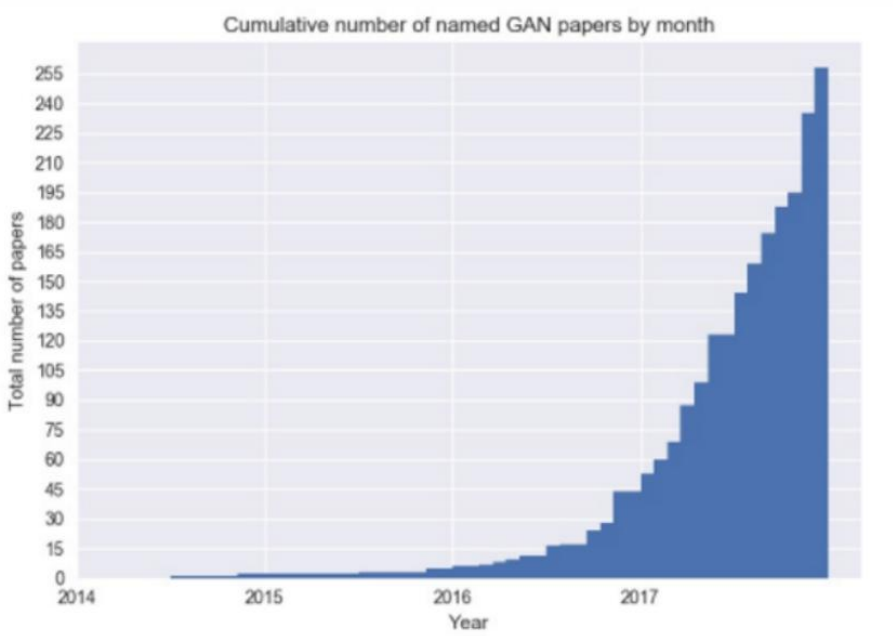
姓名:劉家瑜

學號:M11102150

中華民國 111 年 12 月 28 日

一、摘要

GAN(Generative Adversarial Network)在前幾年與近期都是非常熱門的領域，從下圖歷年論文數量可以觀察出來。



參考自 G. Ian et al., "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.,而這邊我的實作是 DCGAN(Deep Convolutional Generative Adversarial Networks)，顧名思義該神經網路模型採用的是深度捲積網路，由多層捲積層堆疊而成，它的好處在於可以提升 GAN 訓練的穩定性及生成結果的質量。

二、簡介

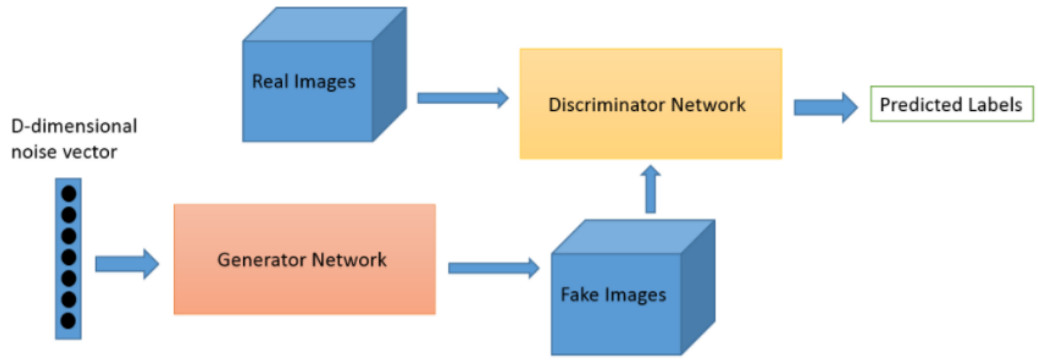
生成對抗網路的概念非常簡單，就像一個遊戲有兩個角色，一個是偽造者(counterfeiter)，他負責製造假鈔，而另一個角色則是警察，需要不斷從偽造者那邊拿到假鈔並判斷究竟是真是假，然後偽造者會根據警察判斷的結果回饋不斷改良假鈔的品質，最後假鈔的品質被改良到和真鈔很相似，因而很難辨別。

硬體規格:

CPU:i7-12700
GPU:RTX 3090

三、設計原理

GAN 分成兩個網路，Generator 跟 Discriminator，而我的期末報告是做人臉生成，因此我的 Generator 要不斷製造出假的人臉，而 Discriminator 則要負責去判斷到底是 fake image 還是 real image，而 Generator 會根據反饋不斷修正並製造 fake image，最後得出來的人臉就像是真的一樣，但其實那個人並不真實存在，一切都只是由 random noise 經過 Generator 製造出來的虛幻人物而已。



Generator model:

```
Generator(
  (main): Sequential(
    (0): ConvTranspose2d(100, 512, kernel_size=(4, 4), stride=(1, 1), bias=False)
    (1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): ReLU(inplace=True)
    (3): ConvTranspose2d(512, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (4): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (5): ReLU(inplace=True)
    (6): ConvTranspose2d(256, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (7): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (8): ReLU(inplace=True)
    (9): ConvTranspose2d(128, 64, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (10): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (11): ReLU(inplace=True)
    (12): ConvTranspose2d(64, 3, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (13): Tanh()
  )
)
```

Discriminator model:

```
Discriminator(
  (main): Sequential(
    (0): Conv2d(3, 64, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (1): LeakyReLU(negative_slope=0.2, inplace=True)
    (2): Conv2d(64, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (3): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (4): LeakyReLU(negative_slope=0.2, inplace=True)
    (5): Conv2d(128, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (6): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (7): LeakyReLU(negative_slope=0.2, inplace=True)
    (8): Conv2d(256, 512, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (9): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (10): LeakyReLU(negative_slope=0.2, inplace=True)
    (11): Conv2d(512, 1, kernel_size=(4, 4), stride=(1, 1), bias=False)
    (12): Sigmoid()
  )
)
```

Dataset:

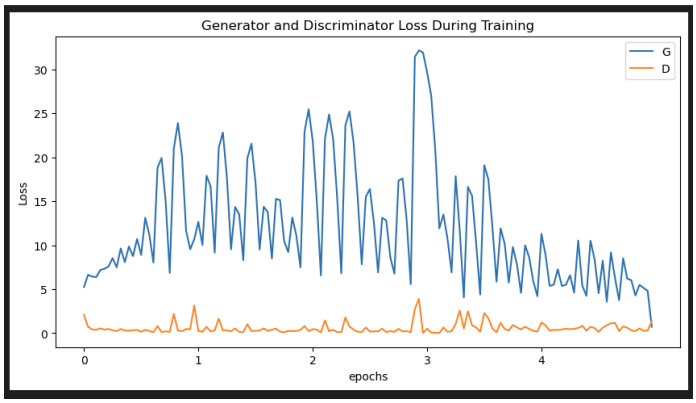
使用網路上的 Large-scale CelebFaces Attributes (CelebA) Dataset，這個資料集共有 202,599 張面部圖像，收錄了 10,177 位知名人物，並且該數據集中涵蓋了巨大的姿態變化和背景雜訊，這個資料集除了拿來訓練 GAN 模型之外，也可以拿來做為臉部辨識或是臉部偵測的訓練或測試資料集，下圖皆為範例圖片。

四、實驗成果

我做了很多組實驗，分別使用不同的 epochs 與不同的 batch size 和 learning rate 並分別把它們記錄下來。

Batch size = 128, Epochs = 5, Learning Rate = 0.0002

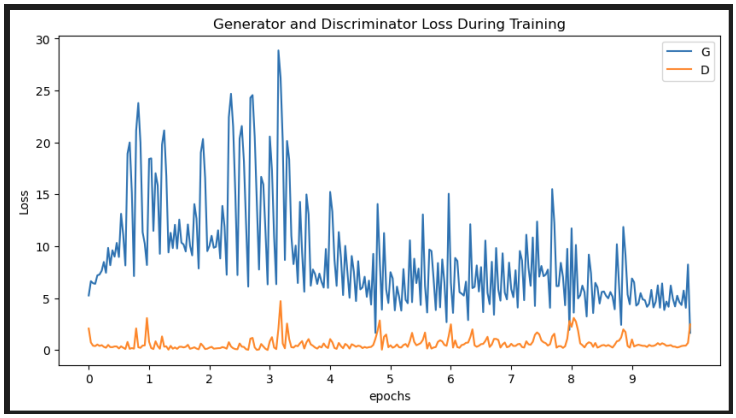
```
Starting Training Loop...
[0/5][0/28]     Loss_D: 2.0722  Loss_G: 5.2568  D(x): 0.4722    D(G(z)): 0.6326 / 0.0078
[1/5][0/28]     Loss_D: 0.2588  Loss_G: 12.6711 D(x): 0.9797    D(G(z)): 0.1855 / 0.0000
[2/5][0/28]     Loss_D: 0.4460  Loss_G: 21.8817 D(x): 0.7986    D(G(z)): 0.0000 / 0.0000
[3/5][0/28]     Loss_D: 0.5167  Loss_G: 29.6986 D(x): 0.8039    D(G(z)): 0.0000 / 0.0000
[4/5][0/28]     Loss_D: 1.2115  Loss_G: 11.2988 D(x): 0.9599    D(G(z)): 0.5839 / 0.0001
```



可以看出經過 5 個 epochs 後，從原本的 random noise 已經漸漸有一點人臉的輪廓。

Batch size = 128, Epochs = 10, Learning Rate = 0.0002

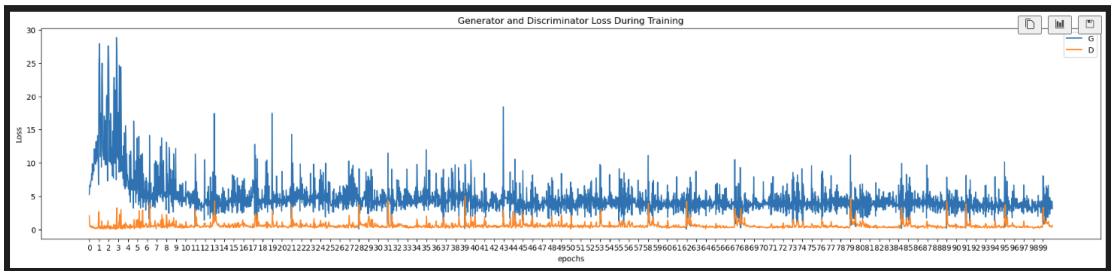
```
Starting Training Loop...
[0/10][0/28]    Loss_D: 2.0722  Loss_G: 5.2567  D(x): 0.4722    D(G(z)): 0.6326 / 0.0078
[1/10][0/28]    Loss_D: 0.8028  Loss_G: 18.4099 D(x): 0.9846    D(G(z)): 0.4647 / 0.0000
[2/10][0/28]    Loss_D: 0.2333  Loss_G: 10.0259 D(x): 0.9677    D(G(z)): 0.1397 / 0.0001
[3/10][0/28]    Loss_D: 0.8398  Loss_G: 20.5674 D(x): 0.9399    D(G(z)): 0.4162 / 0.0000
[4/10][0/28]    Loss_D: 1.0563  Loss_G: 15.2354 D(x): 0.9563    D(G(z)): 0.4299 / 0.0000
[5/10][0/28]    Loss_D: 0.4594  Loss_G: 7.5027  D(x): 0.9790    D(G(z)): 0.3057 / 0.0015
[6/10][0/28]    Loss_D: 2.4790  Loss_G: 6.4899  D(x): 0.2710    D(G(z)): 0.0020 / 0.0136
[7/10][0/28]    Loss_D: 0.6013  Loss_G: 5.8628  D(x): 0.7983    D(G(z)): 0.1771 / 0.0058
[8/10][0/28]    Loss_D: 2.2355  Loss_G: 11.7234 D(x): 0.9178    D(G(z)): 0.7084 / 0.0001
[9/10][0/28]    Loss_D: 1.0205  Loss_G: 6.8938  D(x): 0.9690    D(G(z)): 0.4207 / 0.0054
```

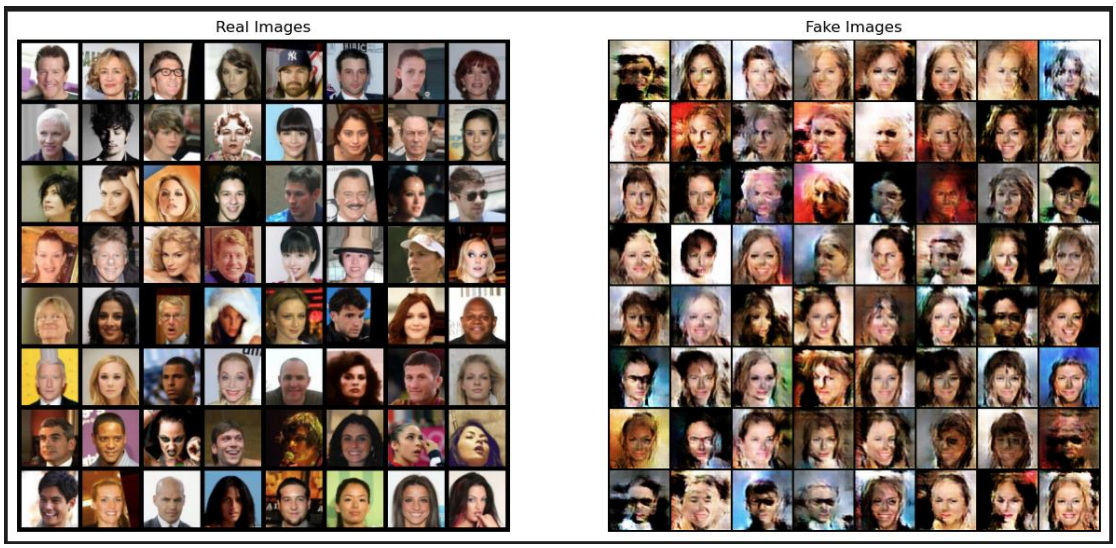


Generator 的 loss 隨著 epochs 數目慢慢下降。

Batch size = 128, Epochs = 100, Learning Rate = 0.0002

```
Output exceeds the size limit. Open the full output data in a text editor
Starting Training Loop...
[0/100][0/28]   Loss_D: 2.0722  Loss_G: 5.2566  D(x): 0.4722    D(G(z)): 0.6326 / 0.0078
[1/100][0/28]   Loss_D: 2.6923  Loss_G: 24.7116 D(x): 0.9891    D(G(z)): 0.8790 / 0.0000
[2/100][0/28]   Loss_D: 0.2255  Loss_G: 25.3161 D(x): 0.8786    D(G(z)): 0.0000 / 0.0000
[3/100][0/28]   Loss_D: 0.2066  Loss_G: 10.5084 D(x): 0.9348    D(G(z)): 0.0153 / 0.0001
[4/100][0/28]   Loss_D: 0.4519  Loss_G: 7.3841  D(x): 0.8803    D(G(z)): 0.1769 / 0.0018
[5/100][0/28]   Loss_D: 1.4655  Loss_G: 6.0187  D(x): 0.4774    D(G(z)): 0.0027 / 0.0108
[6/100][0/28]   Loss_D: 0.2775  Loss_G: 5.7865  D(x): 0.8848    D(G(z)): 0.1001 / 0.0128
[7/100][0/28]   Loss_D: 0.5584  Loss_G: 4.3664  D(x): 0.7380    D(G(z)): 0.0454 / 0.0315
[8/100][0/28]   Loss_D: 2.2249  Loss_G: 13.1141 D(x): 0.9646    D(G(z)): 0.7776 / 0.0000
[9/100][0/28]   Loss_D: 1.3330  Loss_G: 4.4361  D(x): 0.4548    D(G(z)): 0.0093 / 0.0566
[10/100][0/28]  Loss_D: 1.1526  Loss_G: 7.2297  D(x): 0.8753    D(G(z)): 0.5036 / 0.0024
[11/100][0/28]  Loss_D: 3.0837  Loss_G: 11.3218 D(x): 0.9888    D(G(z)): 0.8525 / 0.0001
[12/100][0/28]  Loss_D: 0.9804  Loss_G: 3.7104  D(x): 0.5164    D(G(z)): 0.0291 / 0.1069
[13/100][0/28]  Loss_D: 4.2826  Loss_G: 2.5567  D(x): 0.0880    D(G(z)): 0.0003 / 0.2559
[14/100][0/28]  Loss_D: 1.0059  Loss_G: 5.7475  D(x): 0.9442    D(G(z)): 0.4671 / 0.0217
[15/100][0/28]  Loss_D: 0.6190  Loss_G: 8.0287  D(x): 0.9251    D(G(z)): 0.3562 / 0.0010
[16/100][0/28]  Loss_D: 0.3446  Loss_G: 6.0492  D(x): 0.9319    D(G(z)): 0.1930 / 0.0069
[17/100][0/28]  Loss_D: 0.5325  Loss_G: 5.6548  D(x): 0.7726    D(G(z)): 0.1407 / 0.0102
[18/100][0/28]  Loss_D: 0.5556  Loss_G: 4.1307  D(x): 0.8870    D(G(z)): 0.2521 / 0.0358
[19/100][0/28]  Loss_D: 2.7596  Loss_G: 6.7652  D(x): 0.2153    D(G(z)): 0.0047 / 0.0271
[20/100][0/28]  Loss_D: 1.1223  Loss_G: 8.7262  D(x): 0.9319    D(G(z)): 0.5685 / 0.0009
[21/100][0/28]  Loss_D: 2.7603  Loss_G: 14.3044 D(x): 0.9642    D(G(z)): 0.8699 / 0.0002
[22/100][0/28]  Loss_D: 0.8788  Loss_G: 7.7616  D(x): 0.9328    D(G(z)): 0.4173 / 0.0017
[23/100][0/28]  Loss_D: 0.4654  Loss_G: 4.0891  D(x): 0.7560    D(G(z)): 0.0689 / 0.0569
...
[96/100][0/28]  Loss_D: 0.5529  Loss_G: 5.4246  D(x): 0.9472    D(G(z)): 0.3262 / 0.0122
[97/100][0/28]  Loss_D: 1.2658  Loss_G: 6.6875  D(x): 0.9548    D(G(z)): 0.5331 / 0.0029
[98/100][0/28]  Loss_D: 0.2353  Loss_G: 4.0446  D(x): 0.9533    D(G(z)): 0.1548 / 0.0272
[99/100][0/28]  Loss_D: 3.3328  Loss_G: 8.0434  D(x): 0.9970    D(G(z)): 0.8887 / 0.0026
```



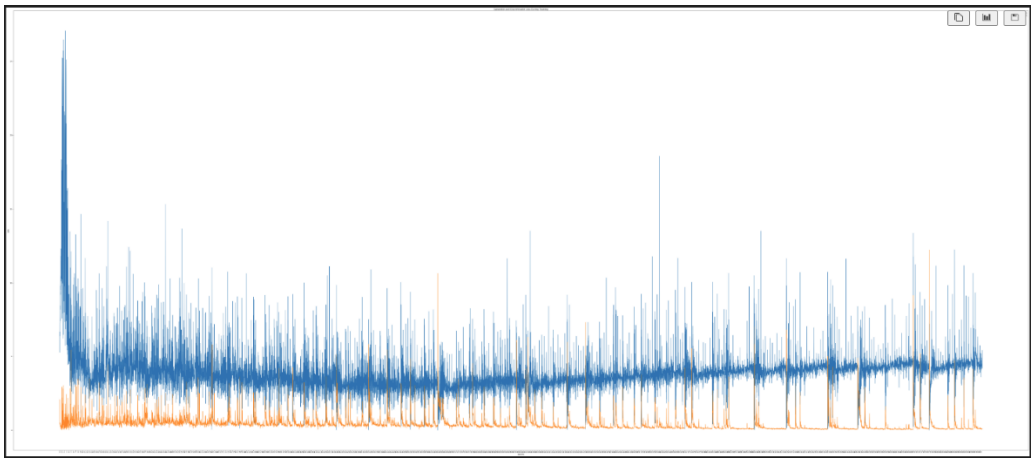


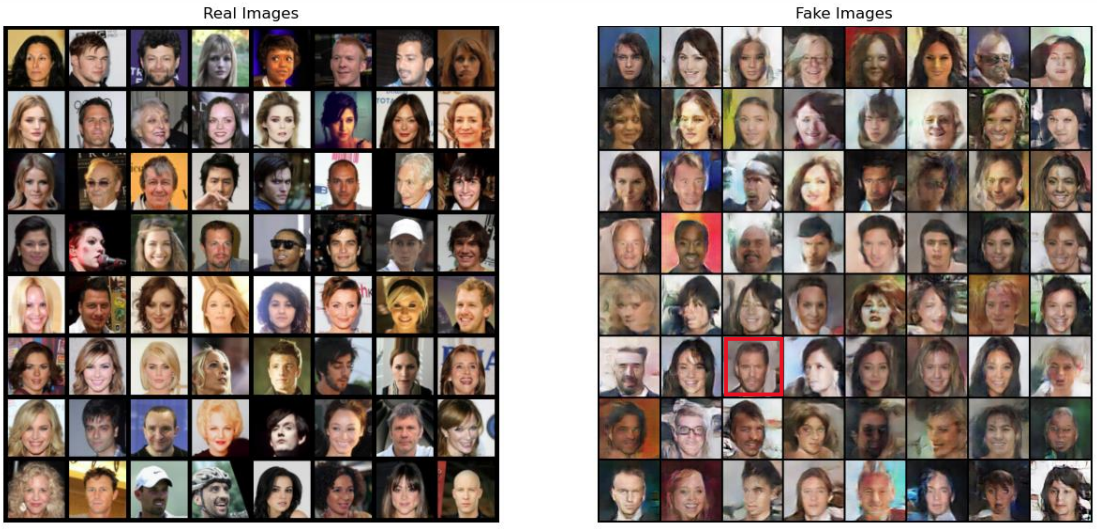
經過 100 次 epochs 後，可以明顯看出 generator 產生的 fake image 基本上已經看得出男女與一些特徵像頭髮顏色等……，但是

五官細節還沒有到非常清楚。

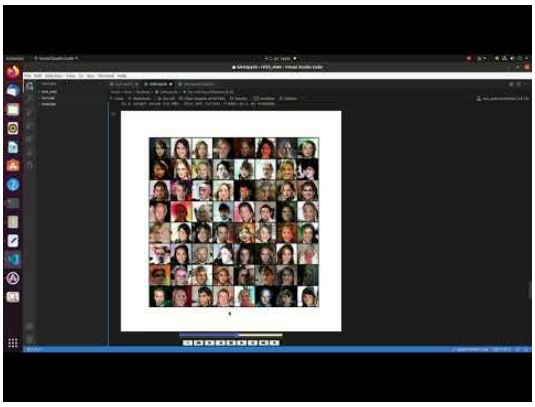
Batch size = 128, Epochs = 400, Learning Rate = 0.0002

```
Output exceeds the size limit. Open the full output data in a text editor
Starting Training Loop...
[0/400][0/28]   Loss_D: 2.0722  Loss_G: 5.2564  D(x): 0.4722    D(G(z)): 0.6326 / 0.0078
[1/400][0/28]   Loss_D: 0.0284  Loss_G: 7.5673  D(x): 0.9793    D(G(z)): 0.0030 / 0.0013
[2/400][0/28]   Loss_D: 0.1725  Loss_G: 7.9746  D(x): 0.9496    D(G(z)): 0.0652 / 0.0017
[3/400][0/28]   Loss_D: 0.3260  Loss_G: 16.2008 D(x): 0.8797    D(G(z)): 0.0001 / 0.0000
[4/400][0/28]   Loss_D: 0.6842  Loss_G: 4.4837  D(x): 0.7180    D(G(z)): 0.0317 / 0.0245
[5/400][0/28]   Loss_D: 0.4974  Loss_G: 1.7328  D(x): 0.7992    D(G(z)): 0.0914 / 0.2864
[6/400][0/28]   Loss_D: 1.1975  Loss_G: 3.5966  D(x): 0.5057    D(G(z)): 0.0487 / 0.1144
[7/400][0/28]   Loss_D: 3.0604  Loss_G: 13.2817 D(x): 0.9808    D(G(z)): 0.9160 / 0.0000
[8/400][0/28]   Loss_D: 1.4099  Loss_G: 8.2310  D(x): 0.9281    D(G(z)): 0.5788 / 0.0009
[9/400][0/28]   Loss_D: 0.5145  Loss_G: 5.8372  D(x): 0.8113    D(G(z)): 0.1676 / 0.0071
[10/400][0/28]  Loss_D: 0.8261  Loss_G: 2.4659  D(x): 0.5842    D(G(z)): 0.0984 / 0.1165
[11/400][0/28]  Loss_D: 1.8333  Loss_G: 11.6687 D(x): 0.9850    D(G(z)): 0.7382 / 0.0001
[12/400][0/28]  Loss_D: 0.3852  Loss_G: 5.1070  D(x): 0.9043    D(G(z)): 0.2025 / 0.0228
[13/400][0/28]  Loss_D: 3.4930  Loss_G: 6.2092  D(x): 0.9836    D(G(z)): 0.8749 / 0.0154
[14/400][0/28]  Loss_D: 2.0229  Loss_G: 7.6743  D(x): 0.9622    D(G(z)): 0.7107 / 0.0108
[15/400][0/28]  Loss_D: 0.7357  Loss_G: 3.5745  D(x): 0.6980    D(G(z)): 0.1941 / 0.0584
[16/400][0/28]  Loss_D: 0.9289  Loss_G: 3.3970  D(x): 0.6167    D(G(z)): 0.1107 / 0.0949
[17/400][0/28]  Loss_D: 0.8026  Loss_G: 5.8738  D(x): 0.9040    D(G(z)): 0.3404 / 0.0082
[18/400][0/28]  Loss_D: 0.5578  Loss_G: 5.2229  D(x): 0.9066    D(G(z)): 0.3180 / 0.0142
[19/400][0/28]  Loss_D: 0.4598  Loss_G: 4.0819  D(x): 0.7930    D(G(z)): 0.1340 / 0.0318
[20/400][0/28]  Loss_D: 0.3599  Loss_G: 4.2459  D(x): 0.8418    D(G(z)): 0.1327 / 0.0257
[21/400][0/28]  Loss_D: 2.3300  Loss_G: 14.1743 D(x): 0.9816    D(G(z)): 0.8243 / 0.0003
[22/400][0/28]  Loss_D: 0.9257  Loss_G: 2.0608  D(x): 0.6276    D(G(z)): 0.1410 / 0.2109
[23/400][0/28]  Loss_D: 0.3272  Loss_G: 2.8012  D(x): 0.8365    D(G(z)): 0.0953 / 0.1146
...
[396/400][0/28] Loss_D: 2.1372  Loss_G: 10.6567 D(x): 0.9991    D(G(z)): 0.7416 / 0.0002
[397/400][0/28] Loss_D: 1.3772  Loss_G: 9.3329  D(x): 0.9766    D(G(z)): 0.5427 / 0.0015
[398/400][0/28] Loss_D: 0.2459  Loss_G: 6.3186  D(x): 0.9856    D(G(z)): 0.1795 / 0.0035
[399/400][0/28] Loss_D: 0.1819  Loss_G: 5.2480  D(x): 0.9912    D(G(z)): 0.1452 / 0.0086
```



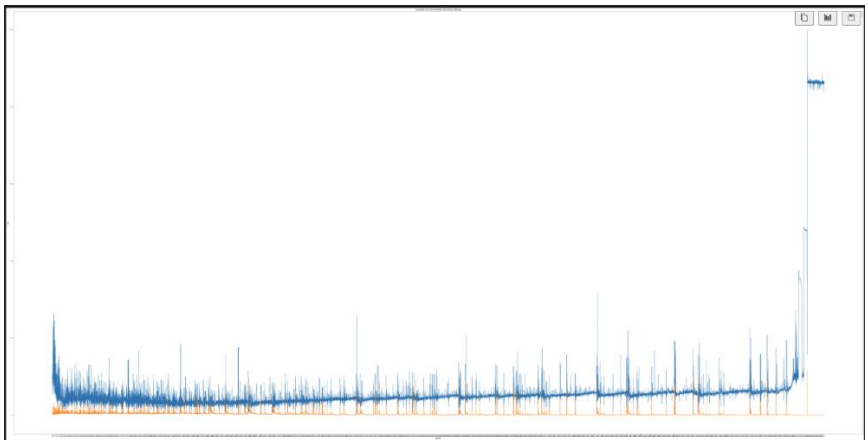


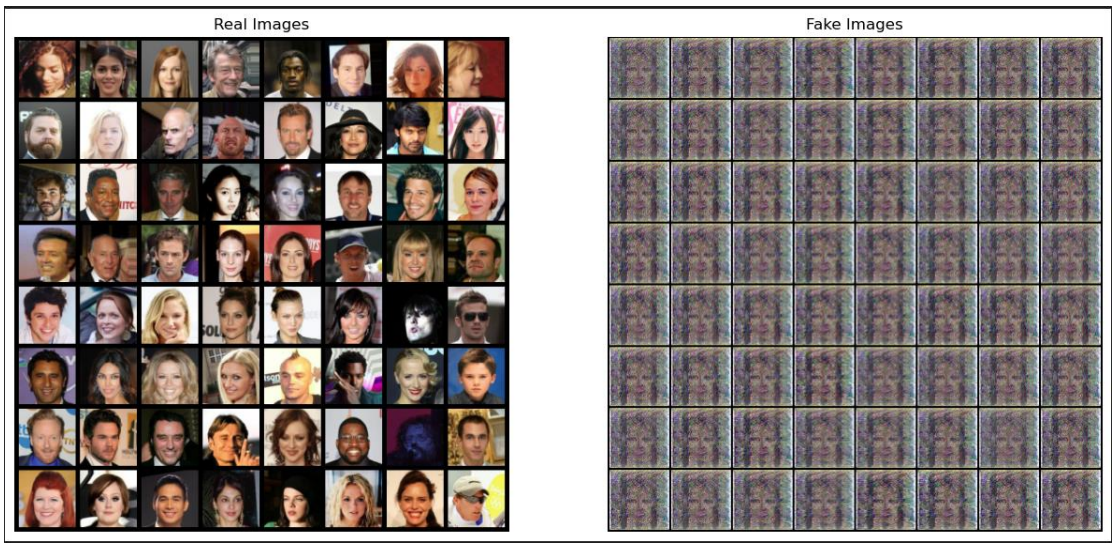
有幾張照片的五官已經非常清晰了，有無戴眼鏡或是有無鬍子，也都能看的出來，可是只有幾張照片而已，有些還是不能精確辨認出該照片的明顯特徵，而且 generator 的 loss 也似乎開始在悄悄上升，這邊有影片紀錄從第 1 個 epoch 到第 400 個的變化過程。



Batch size = 128, Epochs = 800, Learning Rate = 0.0002

```
Output exceeds the size limit. Open the full output data in a text editor
Starting Training Loop...
[0/800][0/28]   Loss_D: 2.0722  Loss_G: 5.2568  D(x): 0.4722    D(G(z)): 0.6326 / 0.0078
[1/800][0/28]   Loss_D: 1.0333  Loss_G: 17.8370 D(x): 0.9936    D(G(z)): 0.5285 / 0.0000
[2/800][0/28]   Loss_D: 0.0695  Loss_G: 9.4241  D(x): 0.9588    D(G(z)): 0.0053 / 0.0004
[3/800][0/28]   Loss_D: 1.6803  Loss_G: 5.3873  D(x): 0.5005    D(G(z)): 0.0001 / 0.0234
[4/800][0/28]   Loss_D: 1.2162  Loss_G: 13.3154 D(x): 0.9445    D(G(z)): 0.4768 / 0.0001
[5/800][0/28]   Loss_D: 0.7829  Loss_G: 6.2318  D(x): 0.8069    D(G(z)): 0.2245 / 0.0082
[6/800][0/28]   Loss_D: 0.5214  Loss_G: 4.7777  D(x): 0.7612    D(G(z)): 0.0959 / 0.0269
[7/800][0/28]   Loss_D: 1.5593  Loss_G: 8.3328  D(x): 0.6082    D(G(z)): 0.3935 / 0.0005
[8/800][0/28]   Loss_D: 1.2727  Loss_G: 8.2785  D(x): 0.9897    D(G(z)): 0.5761 / 0.0009
[9/800][0/28]   Loss_D: 0.3628  Loss_G: 5.0411  D(x): 0.9499    D(G(z)): 0.2123 / 0.0214
[10/800][0/28]  Loss_D: 1.2740  Loss_G: 3.5192  D(x): 0.8066    D(G(z)): 0.3345 / 0.0629
[11/800][0/28]  Loss_D: 3.2811  Loss_G: 6.9405  D(x): 0.9922    D(G(z)): 0.9065 / 0.0086
[12/800][0/28]  Loss_D: 0.6050  Loss_G: 6.2855  D(x): 0.8777    D(G(z)): 0.3242 / 0.0089
[13/800][0/28]  Loss_D: 1.6046  Loss_G: 8.0623  D(x): 0.9760    D(G(z)): 0.6822 / 0.0016
[14/800][0/28]  Loss_D: 1.9976  Loss_G: 9.4141  D(x): 0.9335    D(G(z)): 0.6832 / 0.0015
[15/800][0/28]  Loss_D: 0.5581  Loss_G: 7.3354  D(x): 0.8963    D(G(z)): 0.3266 / 0.0017
[16/800][0/28]  Loss_D: 0.5195  Loss_G: 5.7921  D(x): 0.8118    D(G(z)): 0.1468 / 0.0080
[17/800][0/28]  Loss_D: 0.8322  Loss_G: 4.4292  D(x): 0.7979    D(G(z)): 0.2970 / 0.0300
[18/800][0/28]  Loss_D: 0.8425  Loss_G: 6.0485  D(x): 0.9182    D(G(z)): 0.4190 / 0.0099
[19/800][0/28]  Loss_D: 0.4095  Loss_G: 2.8562  D(x): 0.7618    D(G(z)): 0.0349 / 0.1054
[20/800][0/28]  Loss_D: 0.5023  Loss_G: 7.4408  D(x): 0.9069    D(G(z)): 0.2435 / 0.0026
[21/800][0/28]  Loss_D: 0.7940  Loss_G: 3.1732  D(x): 0.6871    D(G(z)): 0.2008 / 0.1033
[22/800][0/28]  Loss_D: 0.4760  Loss_G: 5.6257  D(x): 0.8855    D(G(z)): 0.2304 / 0.0088
[23/800][0/28]  Loss_D: 0.7638  Loss_G: 5.6807  D(x): 0.8761    D(G(z)): 0.3619 / 0.0110
...
[796/800][0/28] Loss_D: 0.0000  Loss_G: 86.3499 D(x): 1.0000    D(G(z)): 0.0000 / 0.0000
[797/800][0/28] Loss_D: 0.0000  Loss_G: 86.0833 D(x): 1.0000    D(G(z)): 0.0000 / 0.0000
[798/800][0/28] Loss_D: 0.0000  Loss_G: 86.6799 D(x): 1.0000    D(G(z)): 0.0000 / 0.0000
[799/800][0/28] Loss_D: 0.0000  Loss_G: 86.4647 D(x): 1.0000    D(G(z)): 0.0000 / 0.0000
```



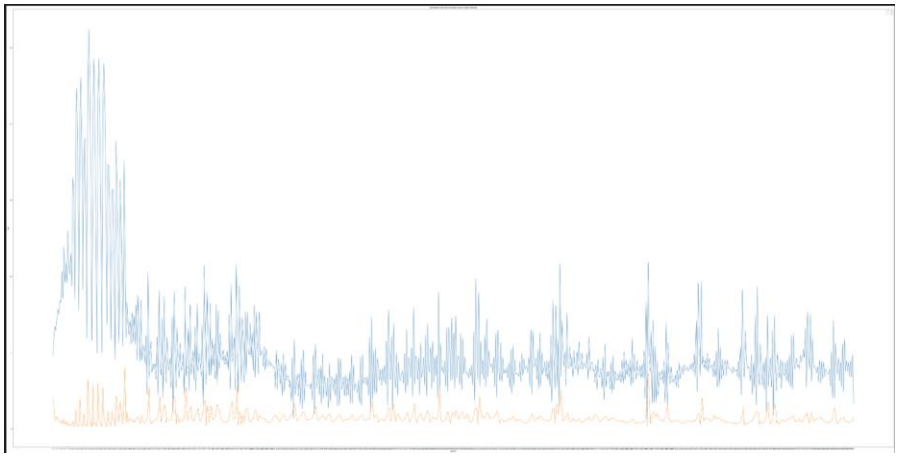


Generator 的 loss 直接爆長，以至於即使 epochs 數目很高也得不

出理想的 fake image，每張圖像甚至有一樣的結果，或許可能是 discriminator 訓練的太好了導致 generator 怎麼樣都會被抓包，因此無法一直進步。

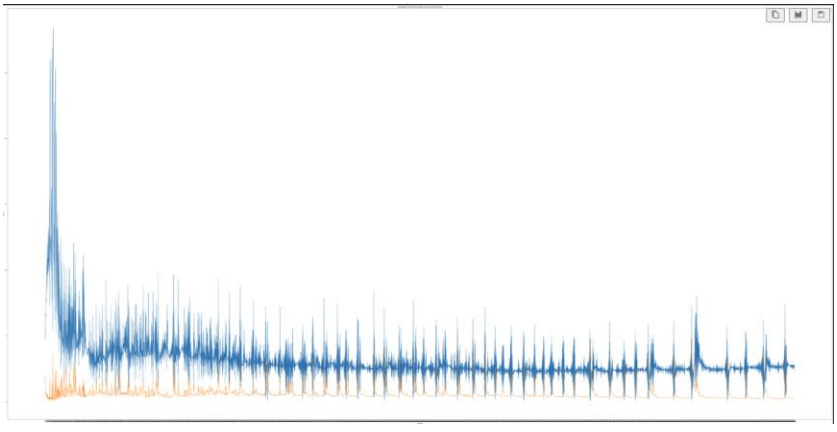
Batch size = 2048, Epochs = 400, Learning Rate = 0.0002

```
Output exceeds the size limit. Open the full output data in a text editor
Starting Training Loop...
[0/400][0/2]    Loss_D: 1.9682  Loss_G: 4.7933  D(x): 0.4662    D(G(z)): 0.5937 / 0.0147
[1/400][0/2]    Loss_D: 0.6391  Loss_G: 6.7177  D(x): 0.9007    D(G(z)): 0.3391 / 0.0024
[2/400][0/2]    Loss_D: 0.8017  Loss_G: 7.1303  D(x): 0.8600    D(G(z)): 0.3611 / 0.0017
[3/400][0/2]    Loss_D: 0.5437  Loss_G: 7.5403  D(x): 0.8432    D(G(z)): 0.1593 / 0.0010
[4/400][0/2]    Loss_D: 0.3911  Loss_G: 8.2853  D(x): 0.8869    D(G(z)): 0.1275 / 0.0005
[5/400][0/2]    Loss_D: 0.2978  Loss_G: 8.5592  D(x): 0.8884    D(G(z)): 0.0550 / 0.0004
[6/400][0/2]    Loss_D: 0.2972  Loss_G: 9.5552  D(x): 0.8683    D(G(z)): 0.0148 / 0.0001
[7/400][0/2]    Loss_D: 0.2524  Loss_G: 9.5718  D(x): 0.9142    D(G(z)): 0.0605 / 0.0001
[8/400][0/2]    Loss_D: 0.2006  Loss_G: 10.4702 D(x): 0.9115    D(G(z)): 0.0092 / 0.0001
[9/400][0/2]    Loss_D: 0.2753  Loss_G: 11.4971 D(x): 0.9356    D(G(z)): 0.1192 / 0.0000
[10/400][0/2]   Loss_D: 0.4758  Loss_G: 16.4621 D(x): 0.9455    D(G(z)): 0.2878 / 0.0000
[11/400][0/2]   Loss_D: 0.1552  Loss_G: 8.5349  D(x): 0.9275    D(G(z)): 0.0026 / 0.0004
[12/400][0/2]   Loss_D: 0.3409  Loss_G: 22.3306 D(x): 0.8404    D(G(z)): 0.0000 / 0.0000
[13/400][0/2]   Loss_D: 0.1360  Loss_G: 7.7663  D(x): 0.9396    D(G(z)): 0.0012 / 0.0009
[14/400][0/2]   Loss_D: 0.3474  Loss_G: 23.0745 D(x): 0.8428    D(G(z)): 0.0000 / 0.0000
[15/400][0/2]   Loss_D: 0.1718  Loss_G: 9.1367  D(x): 0.9185    D(G(z)): 0.0003 / 0.0003
[16/400][0/2]   Loss_D: 0.2682  Loss_G: 19.0750 D(x): 0.8807    D(G(z)): 0.0000 / 0.0000
[17/400][0/2]   Loss_D: 0.1402  Loss_G: 5.8880  D(x): 0.9409    D(G(z)): 0.0115 / 0.0053
[18/400][0/2]   Loss_D: 0.7008  Loss_G: 26.2235 D(x): 0.7120    D(G(z)): 0.0000 / 0.0000
[19/400][0/2]   Loss_D: 0.1807  Loss_G: 14.9547 D(x): 0.9176    D(G(z)): 0.0000 / 0.0000
[20/400][0/2]   Loss_D: 2.8557  Loss_G: 20.9148 D(x): 0.9710    D(G(z)): 0.8997 / 0.0000
[21/400][0/2]   Loss_D: 0.4380  Loss_G: 20.3023 D(x): 0.8064    D(G(z)): 0.0000 / 0.0000
[22/400][0/2]   Loss_D: 0.1610  Loss_G: 4.9632  D(x): 0.9352    D(G(z)): 0.0132 / 0.0119
[23/400][0/2]   Loss_D: 0.5645  Loss_G: 24.3075 D(x): 0.7540    D(G(z)): 0.0000 / 0.0000
...
[396/400][0/2]  Loss_D: 0.4758  Loss_G: 4.4719  D(x): 0.8962    D(G(z)): 0.2580 / 0.0220
[397/400][0/2]  Loss_D: 0.4466  Loss_G: 4.3238  D(x): 0.8491    D(G(z)): 0.2093 / 0.0220
[398/400][0/2]  Loss_D: 0.4434  Loss_G: 4.4555  D(x): 0.8833    D(G(z)): 0.2429 / 0.0194
[399/400][0/2]  Loss_D: 0.5169  Loss_G: 4.8613  D(x): 0.8711    D(G(z)): 0.2786 / 0.0142
```



Batch size = 2048, Epochs = 2000, Learning Rate = 0.0002

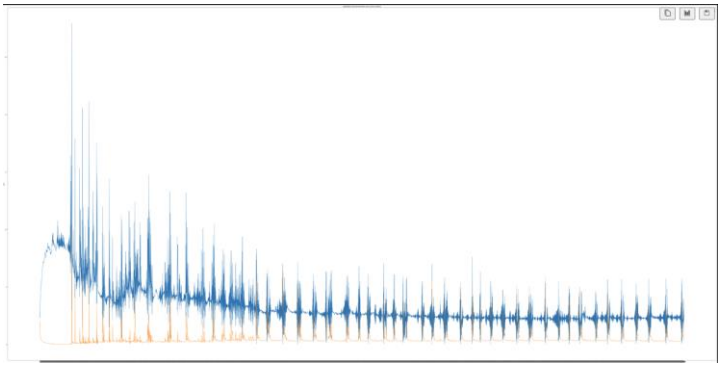
```
Output exceeds the size limit. Open the full output data in a text editor
Starting Training Loop...
[0/2000][0/2]   Loss_D: 1.9682  Loss_G: 4.7932  D(x): 0.4662    D(G(z)): 0.5937 / 0.0147
[1/2000][0/2]   Loss_D: 0.6391  Loss_G: 6.7177  D(x): 0.9006    D(G(z)): 0.3392 / 0.0024
[2/2000][0/2]   Loss_D: 0.8018  Loss_G: 7.1310  D(x): 0.8600    D(G(z)): 0.3611 / 0.0017
[3/2000][0/2]   Loss_D: 0.5441  Loss_G: 7.5415  D(x): 0.8433    D(G(z)): 0.1598 / 0.0010
[4/2000][0/2]   Loss_D: 0.3913  Loss_G: 8.2888  D(x): 0.8868    D(G(z)): 0.1276 / 0.0005
[5/2000][0/2]   Loss_D: 0.2984  Loss_G: 8.5694  D(x): 0.8887    D(G(z)): 0.0560 / 0.0004
[6/2000][0/2]   Loss_D: 0.2982  Loss_G: 9.5819  D(x): 0.8677    D(G(z)): 0.0144 / 0.0001
[7/2000][0/2]   Loss_D: 0.2549  Loss_G: 9.6222  D(x): 0.9152    D(G(z)): 0.0644 / 0.0001
[8/2000][0/2]   Loss_D: 0.1987  Loss_G: 10.2604 D(x): 0.9133    D(G(z)): 0.0111 / 0.0001
[9/2000][0/2]   Loss_D: 0.2095  Loss_G: 9.8538  D(x): 0.9310    D(G(z)): 0.0565 / 0.0001
[10/2000][0/2]  Loss_D: 0.1531  Loss_G: 10.5531 D(x): 0.9224    D(G(z)): 0.0060 / 0.0000
[11/2000][0/2]  Loss_D: 0.1538  Loss_G: 10.0227 D(x): 0.9330    D(G(z)): 0.0142 / 0.0001
[12/2000][0/2]  Loss_D: 0.1849  Loss_G: 13.3643 D(x): 0.9093    D(G(z)): 0.0007 / 0.0000
[13/2000][0/2]  Loss_D: 1.8029  Loss_G: 23.5640 D(x): 0.9623    D(G(z)): 0.7729 / 0.0000
[14/2000][0/2]  Loss_D: 0.3557  Loss_G: 21.3437 D(x): 0.8448    D(G(z)): 0.0000 / 0.0000
[15/2000][0/2]  Loss_D: 0.3162  Loss_G: 12.2869 D(x): 0.9714    D(G(z)): 0.2051 / 0.0000
[16/2000][0/2]  Loss_D: 0.2682  Loss_G: 12.6129 D(x): 0.9568    D(G(z)): 0.1471 / 0.0000
[17/2000][0/2]  Loss_D: 0.4057  Loss_G: 16.2062 D(x): 0.9484    D(G(z)): 0.2390 / 0.0000
[18/2000][0/2]  Loss_D: 0.1546  Loss_G: 7.6244  D(x): 0.9298    D(G(z)): 0.0058 / 0.0010
[19/2000][0/2]  Loss_D: 0.6266  Loss_G: 26.8622 D(x): 0.7164    D(G(z)): 0.0000 / 0.0000
[20/2000][0/2]  Loss_D: 0.1268  Loss_G: 14.0468 D(x): 0.9429    D(G(z)): 0.0000 / 0.0000
[21/2000][0/2]  Loss_D: 3.7469  Loss_G: 25.0590 D(x): 0.9779    D(G(z)): 0.9587 / 0.0000
[22/2000][0/2]  Loss_D: 0.4572  Loss_G: 26.3590 D(x): 0.8025    D(G(z)): 0.0000 / 0.0000
[23/2000][0/2]  Loss_D: 0.0943  Loss_G: 10.4925 D(x): 0.9583    D(G(z)): 0.0001 / 0.0002
...
[1996/2000][0/2]        Loss_D: 0.2356  Loss_G: 2.7955  D(x): 0.9050    D(G(z)): 0.1218 / 0.0801
[1997/2000][0/2]        Loss_D: 0.2380  Loss_G: 2.8468  D(x): 0.9097    D(G(z)): 0.1283 / 0.0757
[1998/2000][0/2]        Loss_D: 0.2408  Loss_G: 2.7624  D(x): 0.9019    D(G(z)): 0.1233 / 0.0830
[1999/2000][0/2]        Loss_D: 0.2221  Loss_G: 2.8421  D(x): 0.9097    D(G(z)): 0.1152 / 0.0762
```

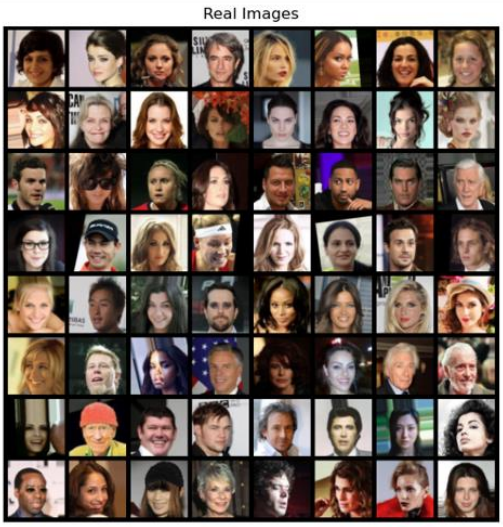


調高 batch size 後，fake image 看起來像是畫作，不過也蠻接近真實圖像的。

Batch size = 4096, Epochs = 5000, Learning Rate = 0.0001

```
Output exceeds the size limit. Open the full output data in a text editor
Starting Training Loop...
[0/5000][0/1]   Loss_D: 1.9760  Loss_G: 2.3245  D(x): 0.4652    D(G(z)): 0.5930 / 0.1428
[1/5000][0/1]   Loss_D: 1.3850  Loss_G: 2.9006  D(x): 0.8608    D(G(z)): 0.6347 / 0.0850
[2/5000][0/1]   Loss_D: 0.9654  Loss_G: 3.5524  D(x): 0.8643    D(G(z)): 0.4847 / 0.0466
[3/5000][0/1]   Loss_D: 0.6933  Loss_G: 3.9343  D(x): 0.8396    D(G(z)): 0.3304 / 0.0323
[4/5000][0/1]   Loss_D: 0.5855  Loss_G: 4.0086  D(x): 0.8367    D(G(z)): 0.2576 / 0.0305
[5/5000][0/1]   Loss_D: 0.5795  Loss_G: 4.0127  D(x): 0.8566    D(G(z)): 0.2722 / 0.0302
[6/5000][0/1]   Loss_D: 0.6023  Loss_G: 4.2193  D(x): 0.8714    D(G(z)): 0.2968 / 0.0251
[7/5000][0/1]   Loss_D: 0.5848  Loss_G: 4.4929  D(x): 0.8723    D(G(z)): 0.2834 / 0.0192
[8/5000][0/1]   Loss_D: 0.5692  Loss_G: 4.7227  D(x): 0.8673    D(G(z)): 0.2638 / 0.0151
[9/5000][0/1]   Loss_D: 0.5261  Loss_G: 4.9066  D(x): 0.8635    D(G(z)): 0.2272 / 0.0124
[10/5000][0/1]  Loss_D: 0.4998  Loss_G: 5.0520  D(x): 0.8678    D(G(z)): 0.2104 / 0.0107
[11/5000][0/1]  Loss_D: 0.4433  Loss_G: 5.1713  D(x): 0.8759    D(G(z)): 0.1789 / 0.0093
[12/5000][0/1]  Loss_D: 0.4118  Loss_G: 5.2883  D(x): 0.8872    D(G(z)): 0.1698 / 0.0084
[13/5000][0/1]  Loss_D: 0.3887  Loss_G: 5.4678  D(x): 0.8960    D(G(z)): 0.1637 / 0.0072
[14/5000][0/1]  Loss_D: 0.3705  Loss_G: 5.6491  D(x): 0.9010    D(G(z)): 0.1563 / 0.0061
[15/5000][0/1]  Loss_D: 0.3542  Loss_G: 5.7845  D(x): 0.9034    D(G(z)): 0.1479 / 0.0054
[16/5000][0/1]  Loss_D: 0.3438  Loss_G: 5.9068  D(x): 0.9049    D(G(z)): 0.1418 / 0.0049
[17/5000][0/1]  Loss_D: 0.3334  Loss_G: 6.0302  D(x): 0.9063    D(G(z)): 0.1364 / 0.0043
[18/5000][0/1]  Loss_D: 0.3215  Loss_G: 6.1789  D(x): 0.9081    D(G(z)): 0.1292 / 0.0039
[19/5000][0/1]  Loss_D: 0.3058  Loss_G: 6.2852  D(x): 0.9109    D(G(z)): 0.1227 / 0.0033
[20/5000][0/1]  Loss_D: 0.2969  Loss_G: 6.4076  D(x): 0.9147    D(G(z)): 0.1220 / 0.0029
[21/5000][0/1]  Loss_D: 0.2894  Loss_G: 6.5998  D(x): 0.9179    D(G(z)): 0.1211 / 0.0025
[22/5000][0/1]  Loss_D: 0.2736  Loss_G: 6.7303  D(x): 0.9202    D(G(z)): 0.1126 / 0.0021
[23/5000][0/1]  Loss_D: 0.2654  Loss_G: 6.8576  D(x): 0.9229    D(G(z)): 0.1099 / 0.0019
...
[4996/5000][0/1]        Loss_D: 0.5228  Loss_G: 3.0659  D(x): 0.9052    D(G(z)): 0.3200 / 0.0641
[4997/5000][0/1]        Loss_D: 0.4406  Loss_G: 2.0831  D(x): 0.7321    D(G(z)): 0.1014 / 0.1625
[4998/5000][0/1]        Loss_D: 0.4049  Loss_G: 2.4865  D(x): 0.8804    D(G(z)): 0.2266 / 0.1076
[4999/5000][0/1]        Loss_D: 0.3614  Loss_G: 2.5139  D(x): 0.8307    D(G(z)): 0.1504 / 0.1066
```





五、心得

我覺得訓練模型是一件很難的事，因為你永遠不知道怎模樣可以得到最好的結果，它中間涉及的變因太多了不管是 optimizer 的選擇或是 learning rate 的調整甚至是模型的組成架構。因此我以 loss 與輸出的結果來調整 model 看看怎樣可以獲得完美的 fake image，據我所知有一個叫做 optuna 的工具可以幫忙找出最適合的 hyperparameters，以後可以使用這個工具，把我得 GAN 模型 train 的更完美，訓練模型的過程除了花時間之外，更重要的是適時調整模型與不斷地觀察 loss 等，看是否有梯度消失或是梯度爆炸甚至是 overfitting 的問題，因此我認為把一個模型訓練得很好，其實不是一件很容易的事。

根據我的實驗，我認為在(Batch size = 128, Epochs = 400, Learning

Rate = 0.0002)和(Batch size = 2048, Epochs = 2000, Learning Rate = 0.0002)的時候生成出的 fake image 挺接近 real image 的。

這裡我提交的 ipynb 檔只有跑過 5 個 epochs 的紀錄，剩下的實驗是用實驗室的 server 來跑的，因為規格好非常多，可以節省非常多的時間。