HW4 Report
姓名：黃浩恩
學號：f05921120
Coworker：賴棹沅、謝忱、嚴聲揚

Q1 Explain the structure of your networks and loss terms in detail (1.5%)

設計兩個 net，分別為 Discriminator(簡稱 D)、Generator(簡稱 G)：

Generator：
因為要透過 G 與 Condition 來生成照片，所以在 G 的設計需要將原本產生的 random 照片與 Condition 兩個資訊當作 G 的輸入，輸出為照片之 array。

1. 先經過 torch.cat 將 random 照片與 Condition 合併。
2. 經過五組 ConvTranspose2d、BatchNorm2d、ReLU 來不斷地降低 feature 維度，擷取 feature，使照片維度從 128*128 降至 4*4。
3. 最後經過 Tanh，將結果傳送回去。

Discriminator：
要透過 D 來驗證 G 是否產生一張好照片，故輸入為照片 array，輸出為[是否為好照片之數值、頭髮之 condition 數值、眼睛之 condition 數值、臉部之 condition 數值、眼鏡之 condition 數值]

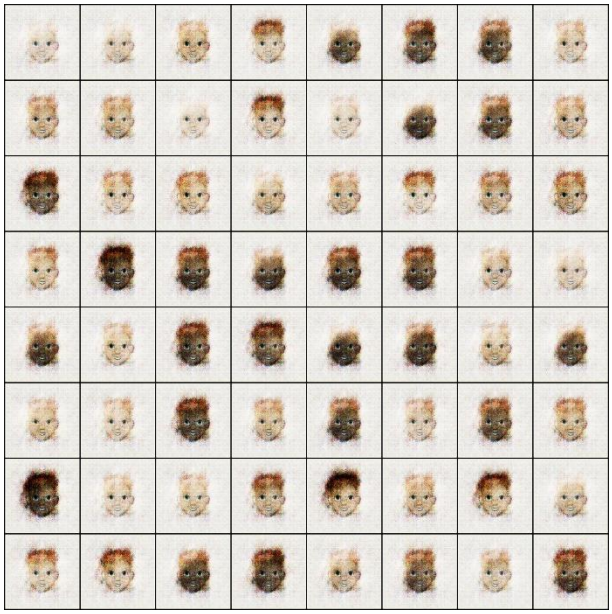
1. 透過五組 Conv2d、LeakyReLU 將小照片做擴維，使 4*4 的照片被展開至 128*128。
2. 建立四個分類器，來分類頭髮、眼睛、臉蛋、眼鏡等資訊，使 D 知曉照片之 feature。
3. 回傳照片評分分數與頭髮、眼睛、臉蛋、眼鏡各類的分數。

最後透過 BCELoss()計算 real 與 fack image、預測頭髮、眼睛、臉蛋、眼鏡與真實 label 的 loss，做最後的更新。

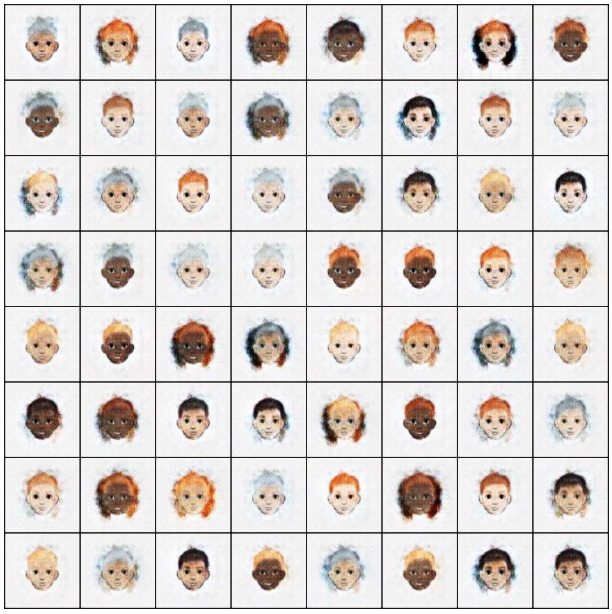
Q2 Plot your training progress (10 pics) (0.5%)

以下之訓練過程為使用 batch==64 之訓練結果：

[1]第 200 個 step：

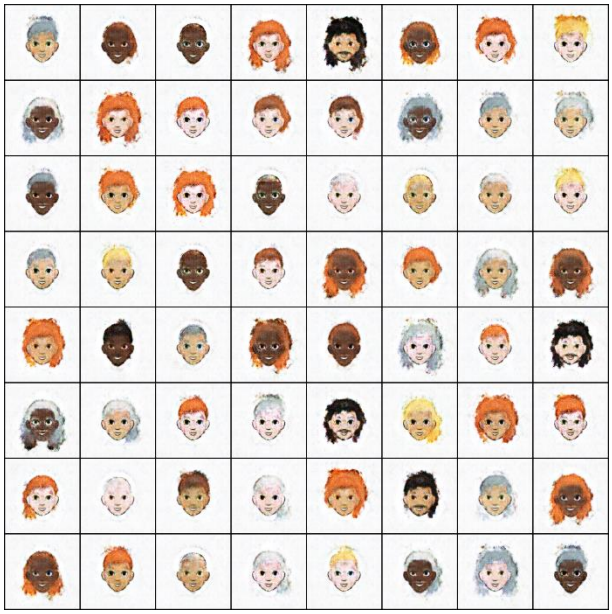


[2]第 400 個 step：

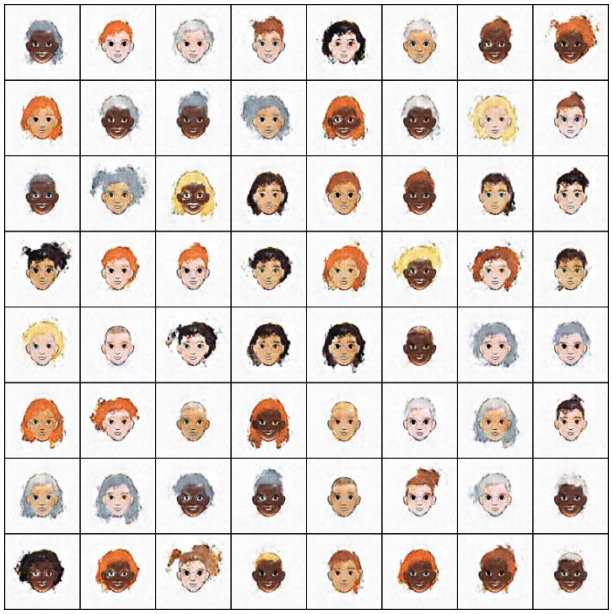


[3]第 600 個 step：

[4]第 800 個 step：



[5]第 1000 個 step：

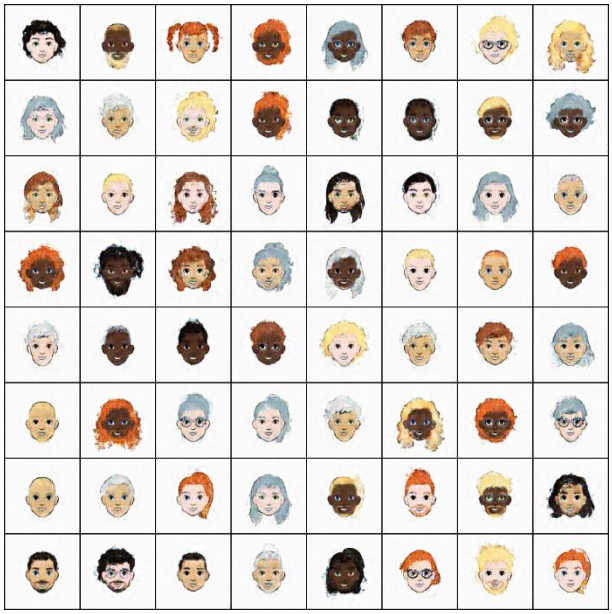


[6]第 2000 個 step：

[7]第 3000 個 step：

[8]第 4000 個 step：

[9]第 8000 個 step：

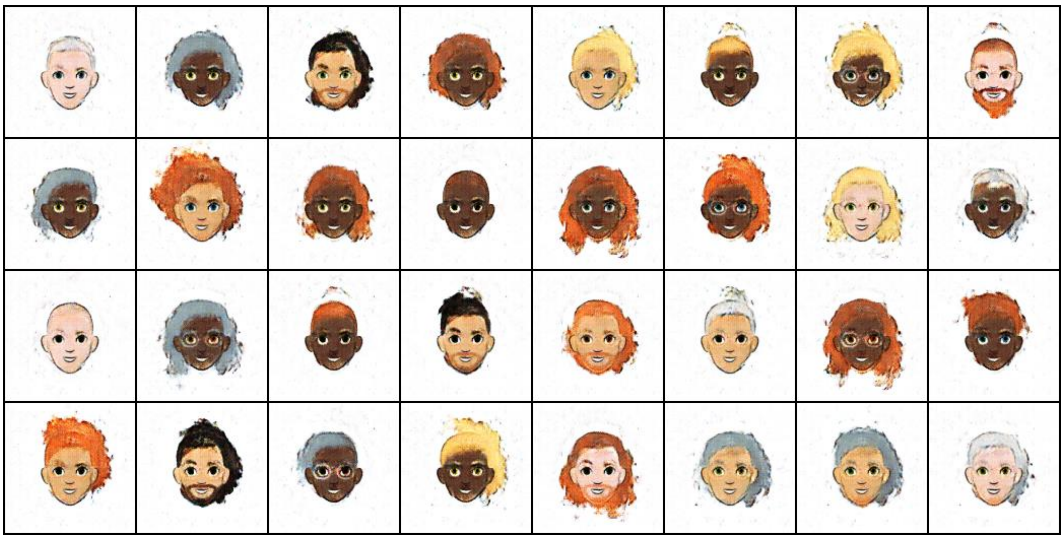


[10]第 10000 個 step：

Q3 Design at least 3 different experiments. Describe your settings, making comparisons and report your observations. (4% + 0.5% bonus)

Q3.1 更改 D 的更新頻率

因為 G 所產生出來的 FID score 都沒有很高，判斷可能 G 還沒 train 起來，D 就太強，所以將 D 的 backward()與 step()的更新頻率下降一倍，使 G 先 train 起來，D 再來判斷。結果如下：



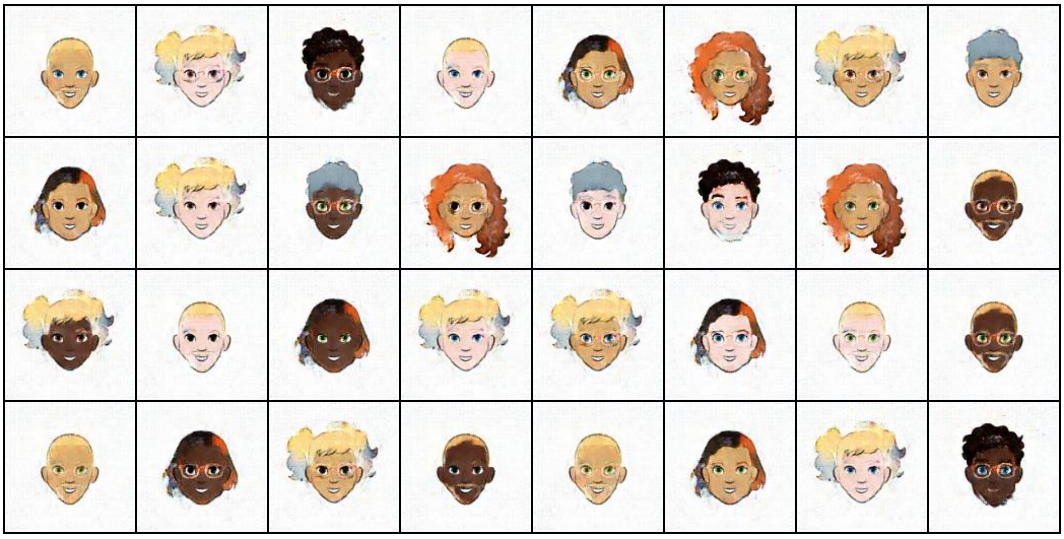
一樣產生出結果，但是圖片看起來髒髒的，FID score 也僅有 221 分，覺得沒有比原先的好。

Q3.2 增加 Classes 算 loss 的權重(classification_weight)

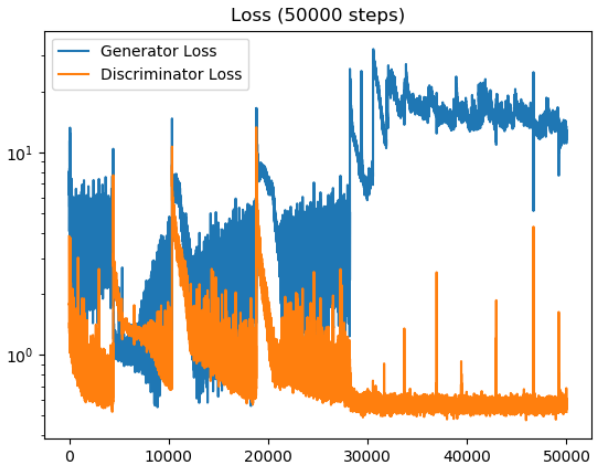
網上許多大神做的 model，在 Condition 出來的 loss 上面都會有加權，讓 model 對各 class 的改進之更新較為敏感，於是嘗試了不同的 weight，有 2 與 10，比較如下：

Q3.2.a Weight=2

更新速度變得比原先要快，可能是因為 Classes 的 loss 加權，使網路知道如何更新，但是過程沒有較好，最後還不斷地爆掉…FID score 也僅有 231 分。

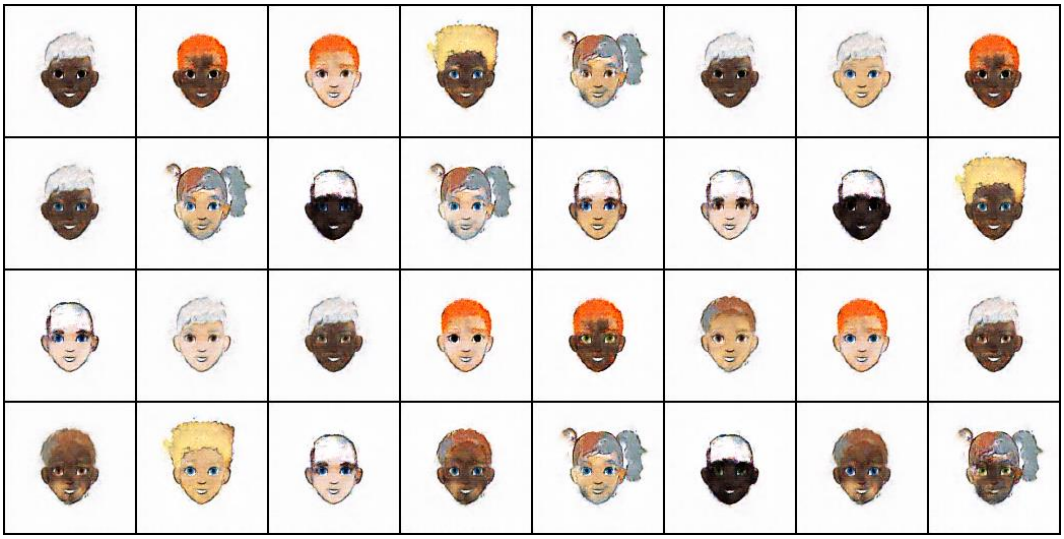


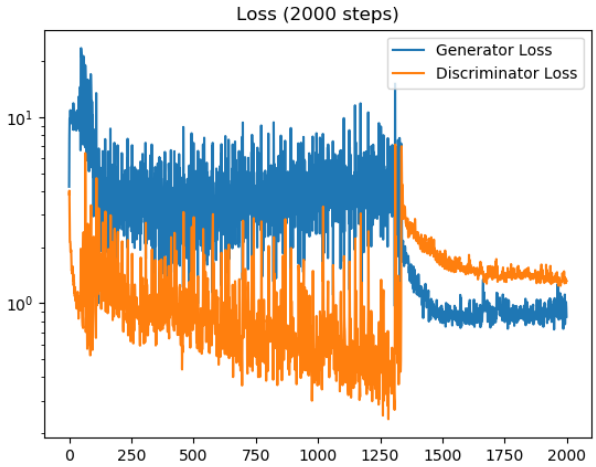
可以從 loss 看到，他不斷地爆掉，最後整個炸了



Q3.2.b　　Weight=10

將 Weight 調成 10，更新速度更快了，但是結果也是沒有變好，從 sample 的途中可以看出，額頭大部分都不見了，索性就把他先停了，也沒有再測 FID score。





Q3.3 更改 latten dim、learning rate，並將 relu 換成 leakyReLu

項目	原先	更改後	FID 結果
latten dim	200	100	166.24→168.19
learning rate	0.0002	0.00005	166.24→166.87
relu	relu	leakyReLu	166.24→166.87

個別測試後，個個狀況皆沒有較多的改善，

Q4 Bonus - Unsupervised Conditional Generation (3%)

於 Bonus 中，使用 infoGAN 來進行 Unsupervised 的訓練，架構為：

Generator：

1. 先將輸入之 noise、labels、code 做 cat，通過一個 Linear，整理 input。
2. 經過三組 BatchNorm2d、Upsample、Conv2d 降維，最後透過 Tanh 輸出預測之照片。

Discriminator：

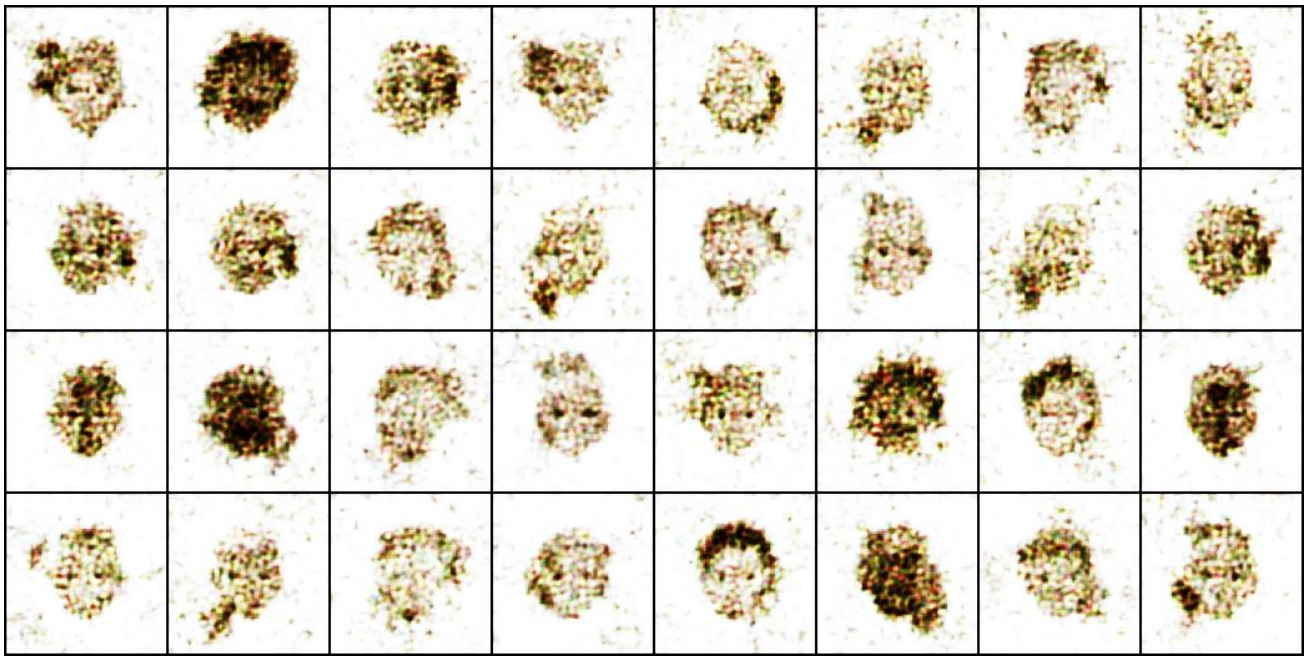
1. 將 image 輸入至 D，經過四組 Conv2d、LeakyReLU、Dropout2d，將照片擴維。
2. 將輸出的結果經過三個不同的 Linear，使判斷照片好壞的分數 fake_pred，與 label、code 可以被評分。

最後經過三個 loss，分別對 adversarial、categorical、continuous 計算 loss，詳細描述如下表：

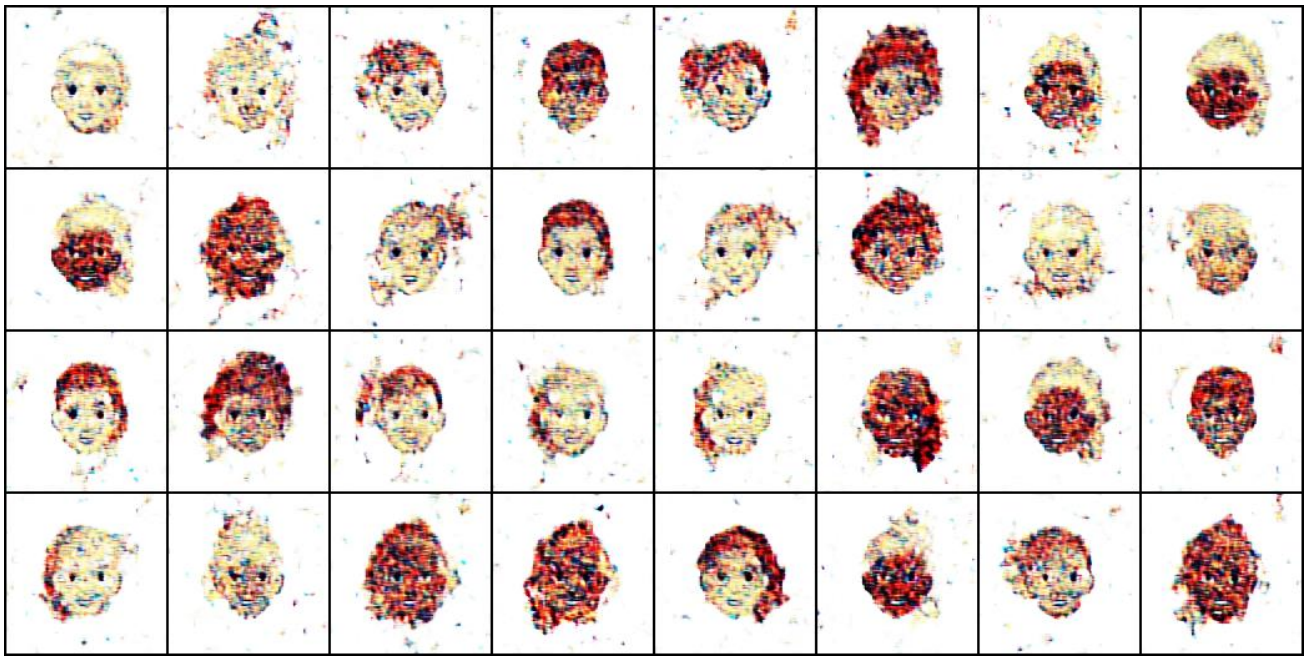
Loss	Method
adversarial_loss	torch.nn.MSELoss()
categorical_loss	torch.nn.CrossEntropyLoss()
continuous_loss	torch.nn.MSELoss()

訓練結果如下：

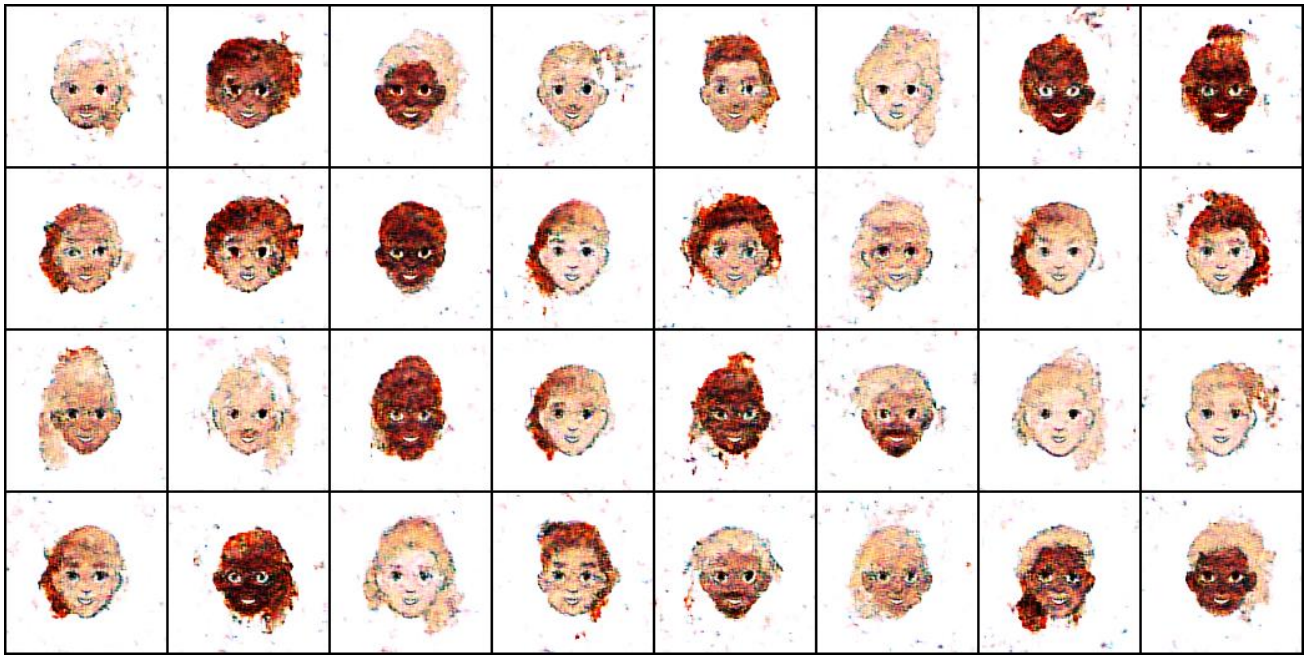
第 1000 個 step



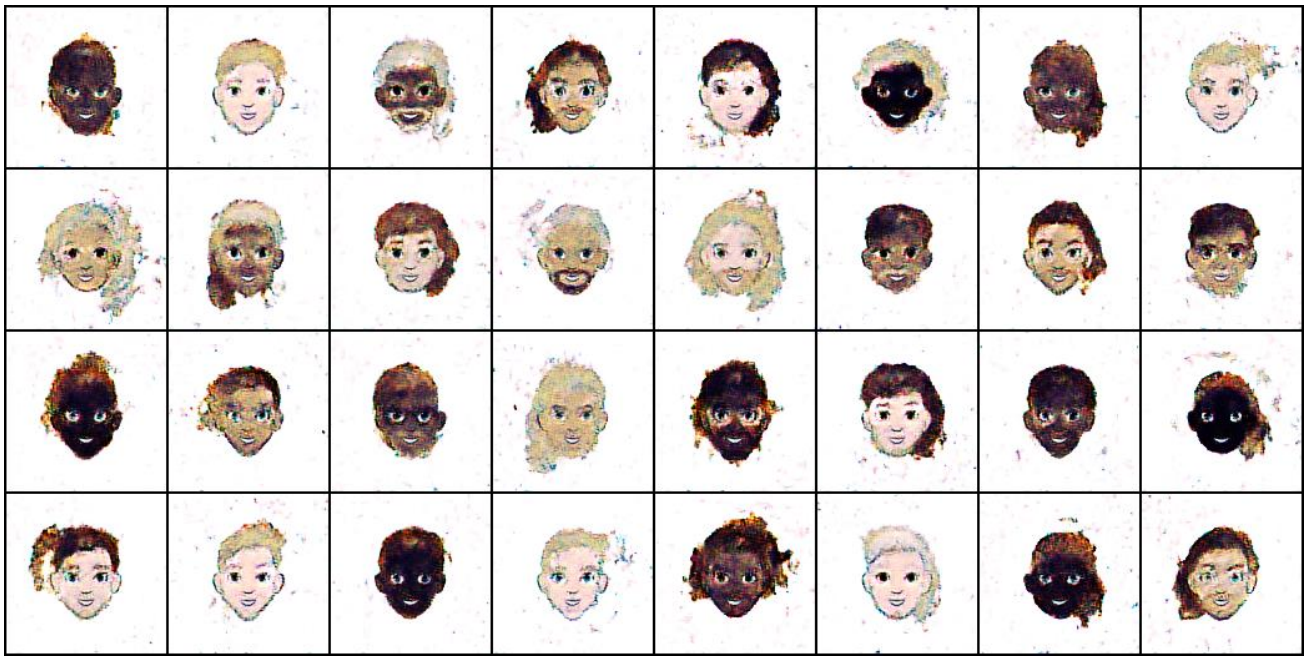
第 5000 個 step



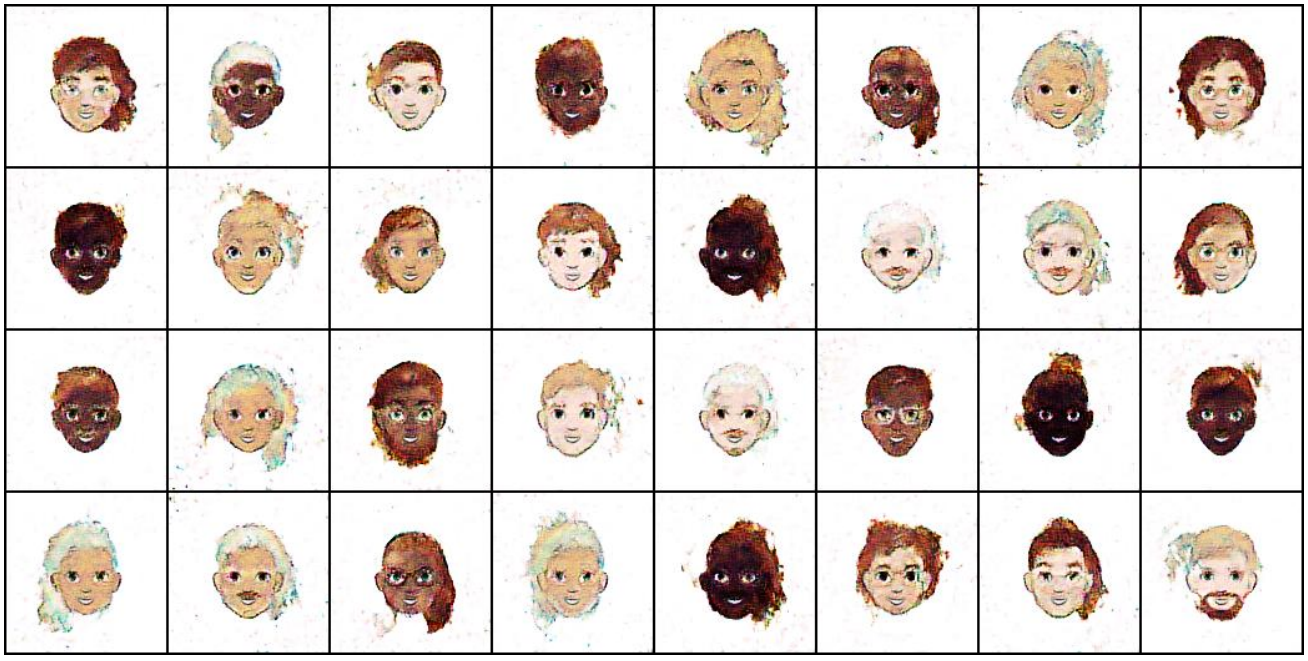
第 10000 個 step



第 20000 個 step



第 30000 個 step



第 4000 個 step