

# MLDS Homework 3 Report

## Model description

### Image Generation

我們使用類似原始 DCGAN 的架構：

Generator 一共有四層 transposed convolution layer，channel size 的大小分別為 256、128、64、3，除了最後一層外，每層的輸出都會經過 batch normalization 後通過 SELU 做 activation，最後的一層通過 tanh 做 activation。整個 generator 的輸入是 100 維的 noise，會先經過線性變換最後轉換成符合輸入的形式 (channel\_size, kernel\_size, kernel\_size)。

Discriminator 則是四層的 convolution layer，channel size 的大小分別為 32、64、128、256，每層的輸出都會經過 batch normalization 以及 SELU 做 activation，第四層的輸出會被拉平經過現性轉換後過 sigmoid 輸出成一個 scalar。

### Text-to-image Generation

我們使用的架構是基於和3-1類似的conditional-DCGAN：

Generator 的部分，hair 以及 eyes 分別以一個數字表示其顏色，兩者分別過一個自己的 embedding layer 變成 5 維，再和 noise 的 100 維 concatenate 後變成 120 維的 input，最後再丟進 3-1 的 generator 架構中。

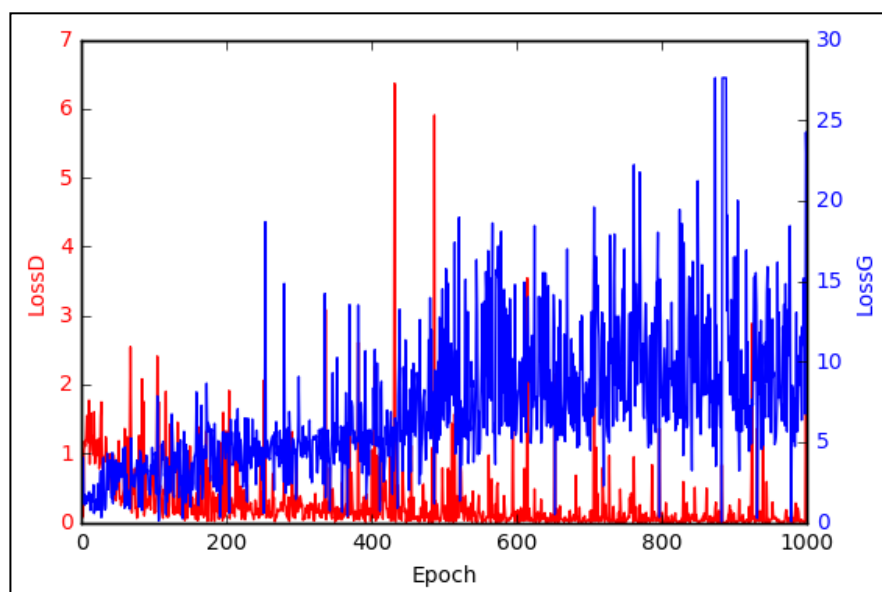
Discriminator 的部分，hair 以及 eyes 一樣分別過一個自己的 embedding 再 reshape 成 (1, 32, 32) 的 1 個 channel 的 tensor，再和 input noise 經過第一個 convolution layer 後的 (16, 32, 32) 的 32 個 channel 的 tensor concatenate 在一起變成 (18, 32, 32) 的 18 個 channel 的 tensor。最後再經過和 3-1 discriminator 後半部一樣的結構。

最後我們還有使用 WGAN 的 clipping 在 (-0.02, 0.02) 之間防止 mode collapse。

## Experiment settings and observation

### Image Generation

在實驗的時候，因為我們用的是 DCGAN，所以要重複的嘗試不同的參數大小，設法讓 Discriminator 與 Generator 的學習過程達到平衡，如果有其中一方太強，Gradient 就會消失。順利的訓練過程如下圖：



觀察 Generator 輸出的圖像，發現 300 - 500 個 epoch 左右時看起來最佳。而其中第 300 個 epoch 的 detection 結果為 25 張人臉 (下圖左)，儘管如此，還是可以看得出來結果不盡人意，雖然輪廓大致正確，但是眼睛大多不太對稱，且有數張較為模糊。而若是繼續訓練至 1000 個 epoch，detection 結果為 19 張人臉 (下圖右)，結果會變的較糟，且可以觀察到嚴重的 mode collapse 的現象。

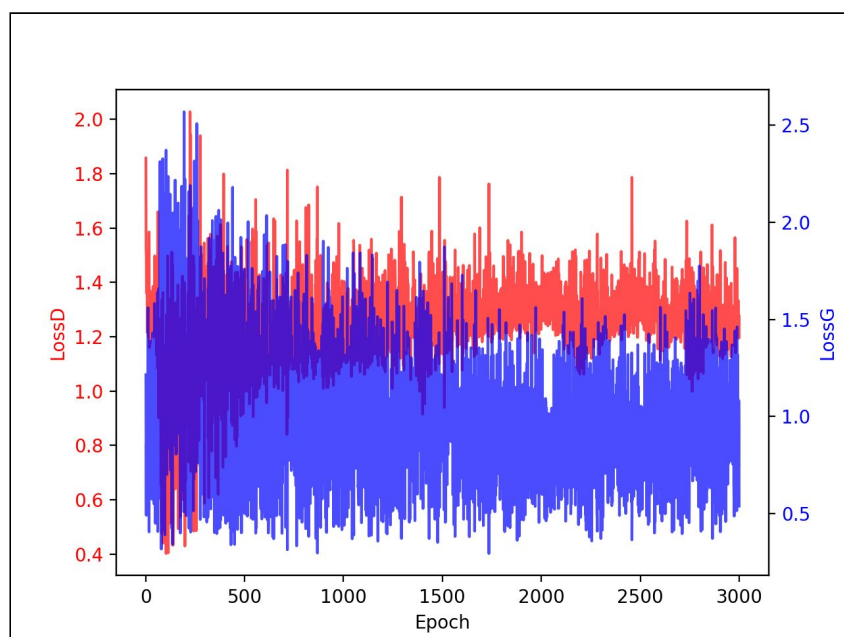


### Text-to-image Generation

Training 過程中，與 3-1 的 dcgan 不一樣的地方是，discriminator 還要判斷如果圖片是正確但是敘述是錯誤的也要判斷他是錯的。因此在 discriminator 中，我們 random 取 hair 以及 eyes 的 label，將其與 real images 做 pair 讓 discriminator 判斷。雖然 random 有可能取到和原始 true label 一樣的值，但是其機率大概只有  $1/(13 \times 11)$ ，因此我們將其忽略。

Generator 我們一開始 embedding 各 50 維再和 noise concatenate，但是如果 embedding 過後的維度太大的話，權重會壓過 noise 造成的差異，很容易導致 mode collapse，因此最後我們 embedding 過後的維度只有 noise 維度的  $1/10$ 。另外在處理 mode collapse 上，我們額外時做了 wgan 的 clipping，讓訓練更加穩定。

訓練過程如下圖，可以看出使用 wgan 的 clipping 可以使 generator 以及 discriminator 的 loss 控制在 2 以下，讓訓練更穩定，不會出現 loss 突然暴增的情況。



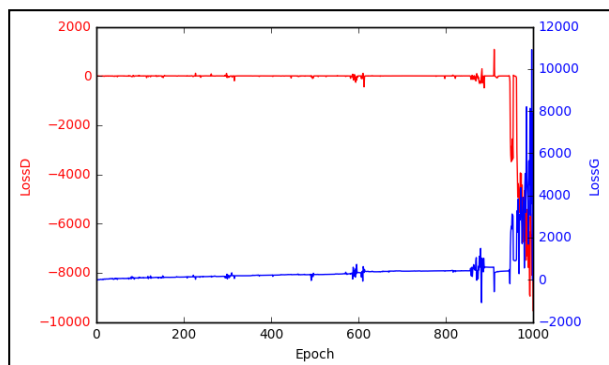
結果如下圖，由上至下依序為藍髮藍眼、藍髮綠眼、藍髮紅眼、綠髮藍眼、綠髮紅眼。由結果可以看出頭髮以及眼睛c-dcgan都能生出正確的顏色，且用baseline.py能夠判斷出25張臉。即使如此，以肉眼來看臉型扭曲以及眼睛只有生成一隻的情況比第一題的純dcgan還要嚴重。我們有試著noise size調整至400、noise sample std調小至0.432、增加generator或是discriminator的channel數，結果都如下圖一樣沒辦法產生更清晰的頭像。



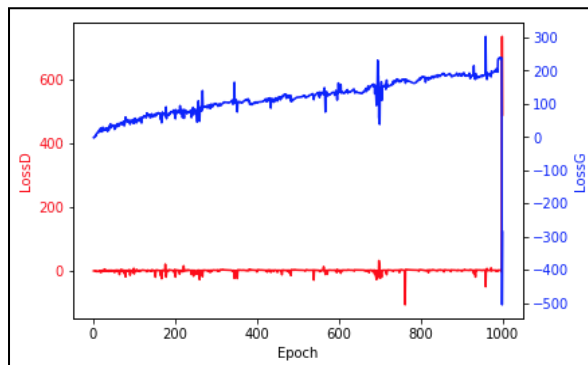
## Compare DCGAN with WGAN-GP

### Image Generation

我們選擇用 WGAN GP 來比較與 DCGAN Image Generation 的結果以及訓練過程，模型的結構大致與 DCGAN 一樣，但是 WGAN GP 需要修改 Discriminator 以及 Generator 的 Loss。我們設置不同的權重 ( $\lambda$ ) 的 gradient penalty，觀察是否影響訓練。



**lambda = 1**



**lambda = 10**

我們發現在 WGAN GP 的訓練過程中，Generator 的 Loss 會一直持續上升，有可能是因為對抗的過程中，同樣的 Update 次數要讓 Generator 騙過 Discriminator 變得越來越難，因此我們猜測如果把 Generator 的 Update 次數再調高的話可以放緩 Generator Loss 上升的幅度。另外我們也發現，當把 lambda 條高至 10 時，Generator Loss 上升的幅度也會趨緩。總而言之，在訓練過程中，合適的 lambda 可以讓訓練更穩定。但是 WGAN GP 還是會出現 Loss 值劇烈變化的情況，在 lambda = 1 的實驗時 1000 Epoch 時 Discriminator 的強度遠遠蓋過 Generator。而 lambda = 10 時則相反。

接著觀察產生出來的圖像，我們取 300 epoch 進行比較：



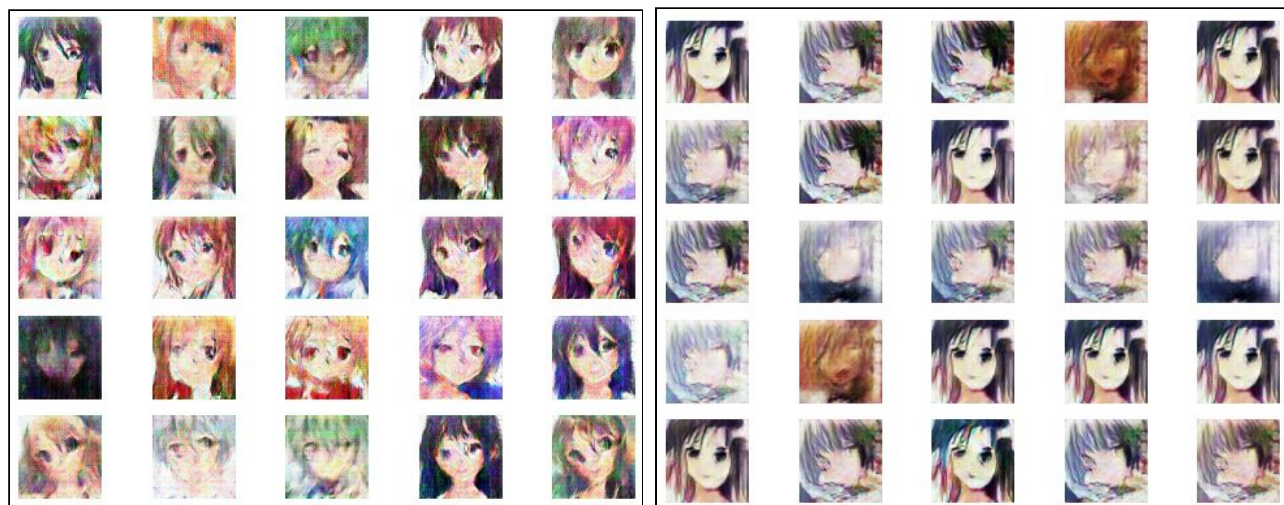
**lambda = 1**



**lambda = 10**

結果看起來並沒有比 DCGAN 生成的更佳。不過若將第 800 epoch 的結果拿出來比較，WGAN GP 看起來比較沒有 model collapse 的現象，結果如下。





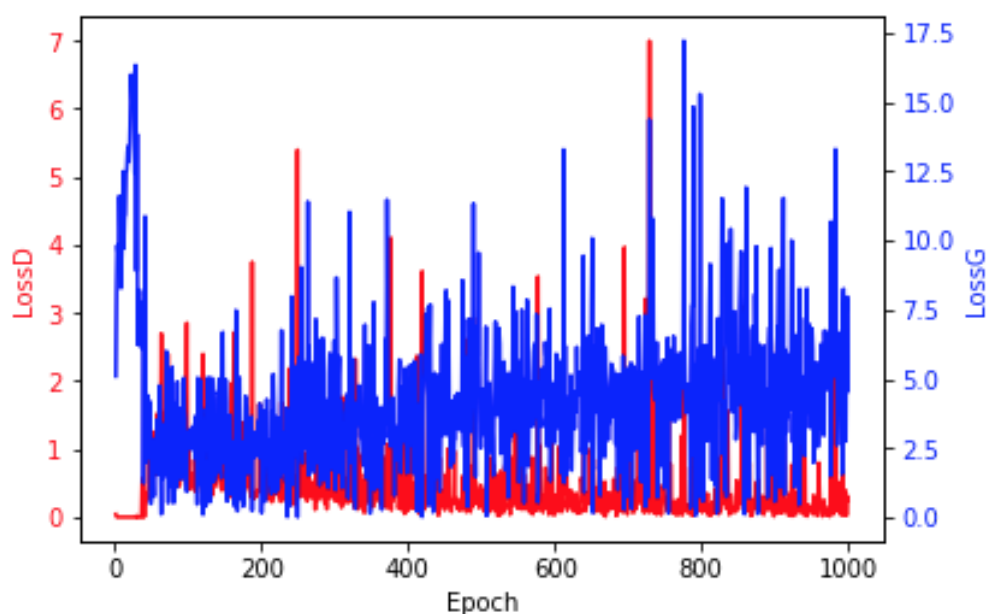
WGAN GP lamda = 10

DCGAN

## Training tips for improvement

### Normalize the Input

我們在 DCGAN 的訓練中有將 0-255 的像素值 normalize 到 -1 至 1 之間，並且使用 Tanh 做為 Generator 最後的 activation function。所以為了觀察沒有 Normalize 的情況，我們做了一次不 normalize 到 -1 至 1、最後的 activation function 為 relu 的實驗。訓練過程如下圖。

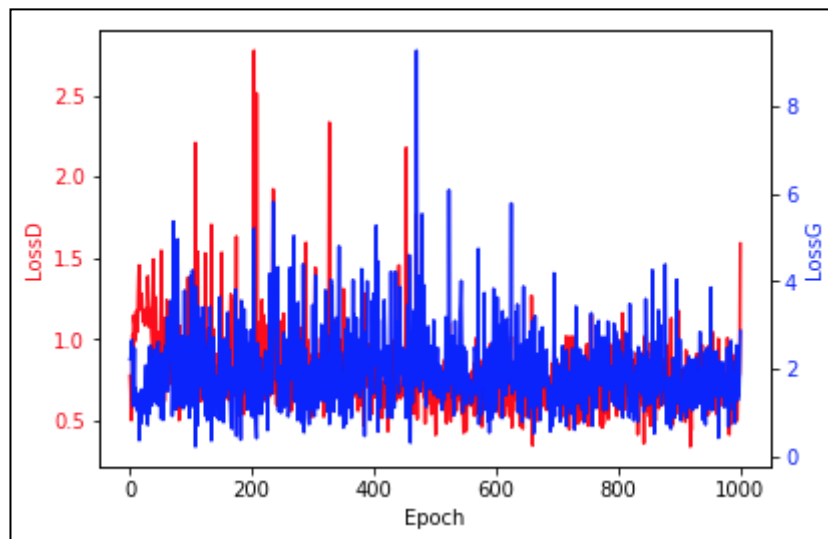


看起來 Generator 與 Discriminator 都有正常的進行訓練，跟 DCGAN 沒什麼分別，但是實際上把輸出的結果打開來看，不管在哪個 epoch 卻都是一片黑。我們推測會出現這樣的情況可

能是雖然 Generator 與 Discriminator 都有在更新但是兩者學習的速度都太慢了。

### Use Soft and Noisy Labels

我們的 DCGAN 沒有使用 soft label，但若用同樣 DCGAN 的架構實驗，真的 label 取樣自 0.7 至 1.2 的值域，假的 label 則取樣自 0.0 至 0.3。訓練的結果如下：

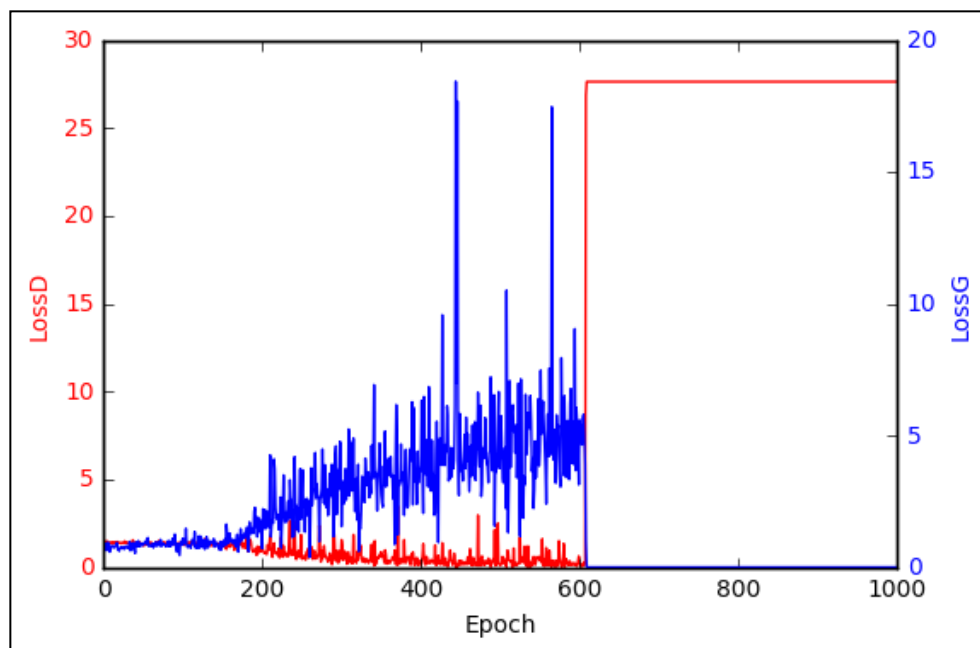


看起來跟原本沒有太大的差別。第 300 epoch 的輸出結果如下，並沒有明顯的進步。



### Batch Normalization

我們的 DCGAN 中有使用 Batch Normalization，若將 Generator 以及 Discriminator 中的 Batch Normalization 層全部移除，再進行實驗。訓練的過程如下圖：



一樣觀察 300 至 500 epoch 的結果，其中第 500 個 epoch 的 detection 結果最好，為 23 張臉 (如下圖)。圖片品質的差距並沒有很大。不過 600 個 epoch 以後 Generator 的 Loss 就歸零無法再更新了。



## Style Transfer



在這部分我們使用 StarGan 做為我們的模型來進行變化，從左到右依序是原圖、黑髮、金髮、棕髮、男性、年輕的變化。從這之中，會發現關於髮色變化，平均來說 StarGan 處理還算不錯，而且依據不同的髮色，臉的膚色也會跟著做適度的修改。然而，像是最後一排的變化卻不是那麼理想，我們認為這主要是因為原本照片比較模糊再加上沒有修正原圖的光線平衡，才會讓 StarGan 較難產出理想的結果。此外，在第四排中，會發現透過 StarGan 產生變化時，人的衣著顏色也跟著有所變化，我們認為這主要是因為在 CelebA 中的圖片都是以特寫人臉為主，比較少有衣著的部分，因此在訓練時模型並不會因此訓練到不能更改衣著的顏色，才會造成衣著顏色改變的現象。

分工表	
r06725008 郭毓棠	3-3
r06725005 郝思喬	3-2
r06725020 劉冠宏	3-1