

MLDS HW3

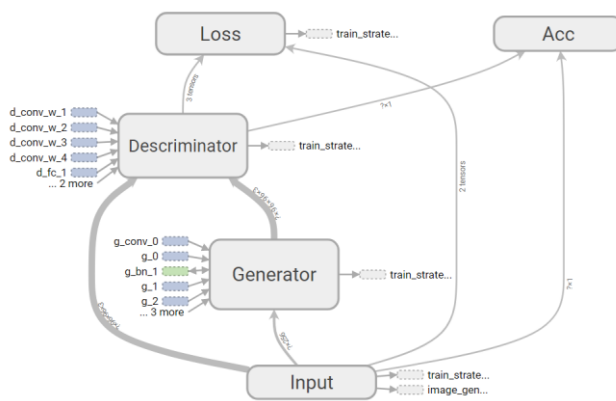
R06921062 蘇楷鈞、R06922104 黃鈞宥、R06921026 林羿辰

I. Model Description

1. Image Generation

我們的 model 基本以 WGAN-GP 實作

Model structure



Model Setting

Generato 的 Activation 使用 selu

learning rate = 2e-4 with RMSprop

Objective function :

$\text{Min} (-\text{Max}(D(x)) - \text{Min}(y * \log(\text{Clf}(x)))$

Discriminator 的 Activation 使用 relu

learning rate = 2e-4 with RMSprop

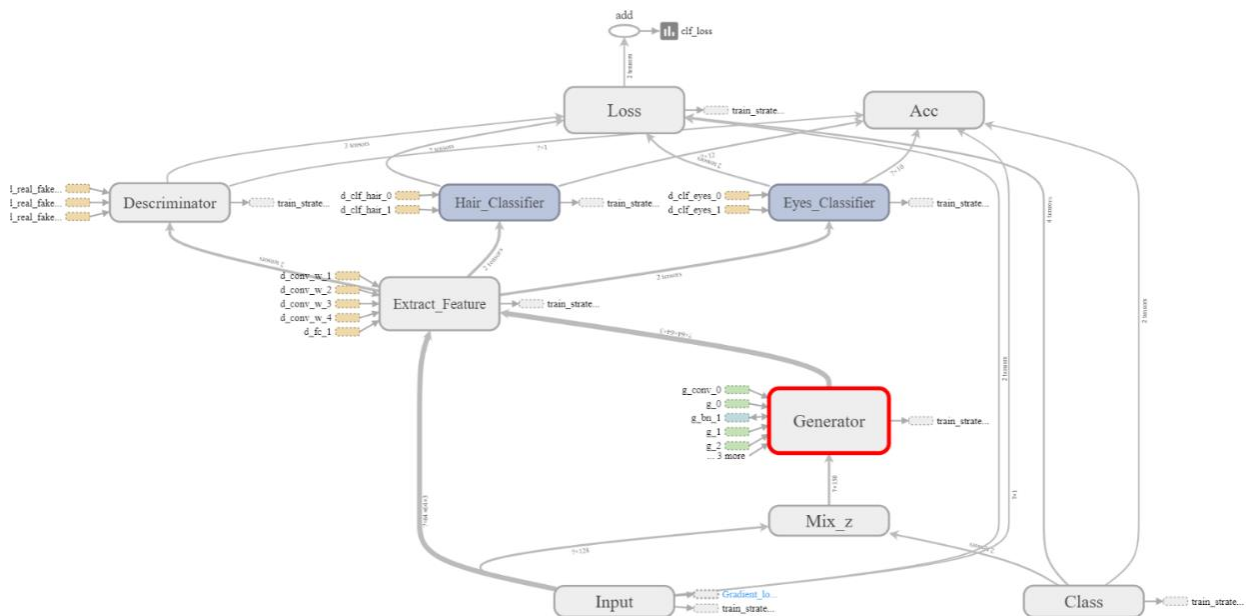
Objective function : $\text{Min}(-D(x) + D(\tilde{x})), x \text{ is real image }, \tilde{x} \text{ is fake image}$

Model hyperparameter

| Generator | | Discriminator | |
|--|--|--|--|
| Layer name (kernel size) / Filters / strides | | Layer name (kernel size) / Filters / strides | |
| Fully-connected / 16*16*128 | | Conv2D(8x8)/64/8 | |
| Reshape(16,16,128) | | avg_pool2d (3,3) / - / 2 | |
| Conv2D_Transpose/ 128 / 2 | | Conv2D(8x8)/64/6 | |
| Conv2D/128 / 4 | | avg_pool2d (3,3) / - / 2 | |
| Conv2D_Transpose / 64 / 2 | | Conv2D(8x8)/128/4 | |
| Conv2D / 3 / 4 | | avg_pool2d (3,3) / - / 2 | |
| tanh | | Conv2D(8x8)/256/4 | |
| | | avg_pool2d (3,3) / - / 2 | |
| | | Flatten | |
| | | Fully-connected / 1024 | |
| | | Fully-connected / 1024 | |
| | | Fully-connected / 1 | |

2. Text-to-image Generation

Model structure



Model setting

Generator 的 Activation 使用 selu Adam(1e-4)

Objective function: $\text{Min}(-\text{Max}(D(x)) - \text{Min}(y * \log(\text{Clf}(x))))$

Discriminator 的 Activation 使用 leaky_relu RMSProp(1e-4)

Objective function: $\text{Min}(-D(x) + D(\tilde{x})), x \text{ is real image}, \tilde{x} \text{ is fake image}$

Classifier 的 Activation 使用 leaky_relu Adam(1e-4)

Objective function: $\text{Min}(-y * \log(D(x))), x \text{ is real image}, y \text{ is ground truth of image}$

Model hyperparameter

| | Layer name (kernel size) / Filters / strides |
|------------------|--|
| Generator | Conv2D (1x1) / 2048 / 1 |
| | Reshape(2,2,512) |
| | Conv2D_Transpose (4x4) / 256 / 2 |
| | BatchNormalization |
| | Conv2D_Transpose (4x4) / 128 / 2 |
| | Conv2D_Transpose (9x9) / 64 / 4 |
| | BatchNormalization |
| | Conv2D_Transpose (5x5) / 32 / 2 |
| | Conv2D (1x1) / 3 / 1 |
| | tanh |

| | | | Layer name (kernel size) / Filters / strides | | |
|-------------------------------|--------------------------------|--------------------------------|--|----------|---------|
| Extractor | | | Conv2D (5x5) / 256 / 2 | | |
| | | | Conv2D (3x3) / 192 / 1 | | |
| | | | avg_pool2d (3,3) / - / 2 | | |
| | | | Conv2D (3x3) / 128 / 1 | | |
| | | | Conv2D (3x3) / 64 / 2 | | |
| | | | Flatten | | |
| | | | Fully-connected / 256 | | |
| | | | Desc | Clf hair | Clf Eye |
| Fully-connected / 64 | Fully-connected / 12 / softmax | Fully-connected / 10 / softmax | | | |
| Fully-connected / 1 / sigmoid | | | | | |

II. Experiment settings and observation

1. Image Generation

在訓練 model 之前，我們先檢查了原本訓練的資料集，發現裡面有許多不是人臉的圖片，我們使用了與 baseline 中相同的 lbpcascade_animeface[1]的模型將所有非人臉的圖片與不像動漫的人臉圖片給挑出來。總共挑出了 3030 張的圖片不使用。將這些圖片挑出來之後，訓練的圖片結果比較好。

為了得知我們做出來的模型好壞，我們參考了 Progressive Growing of GANs for Improved Quality, Stability, and Variation[2]的兩種評量 GAN 模型的方式。

第一種是以 MS-SSIM 演算法判斷生成圖片的相似程度，共抽出一萬張生成圖片做比較，如果數字較高，代表可能發生了 mode collapse 現象。

第二種是以 Sliced Wasserstein distance 去計算在生成圖片空間與原始圖片空間的相似程度。概念上是將圖片以 SIFT 抽出特徵後，計算生成圖片空間抽取出來的特徵與原始圖片空間抽取出來的特徵之間的距離，並考慮了將圖片縮小在不同尺度時的相似程度。我們在生成圖片與原始圖片分別抽了一萬張圖片來做為評分。數字越小代表生成圖片與原始圖片越相似。

我們的 model 基本以 WGANP 實作 相較於一般 GAN，WGANP 的確比較不會 mode collapse, 細節也相對處理的比較好，可是有可能發生臉部結構不夠完整或變形，眼睛在臉頰的狀況，其中眼睛在臉頰的情形是 wgan 或是 wgangp 特有的狀況，但是這不是調整 Lambda 能夠修正的, lambda 只會縮放 Discriminator 的估值, 同 weight clipping, 具體來說 dicriminator 的重要程度比較高,能夠比較好指引 generator, 而 wgangp 不會 mode

collapse 的特性 也比較能夠增加 training 的 iteration,來收斂細節



| MS-SSIM | SWD-16 | SWD-32 | SWD-64 | SWD-Avg |
|---------|---------|---------|---------|---------|
| 0.3111 | 23.5629 | 20.2422 | 64.6383 | 36.1578 |

2. Text-to-image Generation



可看出都有符合類別的描述，且就算固定類別，在不同的 latent vector 所表現出來的頭髮顏色也不一定一致，可見 generator 是有學到某個範圍就可辨別出某個顏色，而非一對一的變化；眼睛的大小、角度，也不太影響眼睛顏色類別的正確性，例如側臉還是能正確填入顏色；而唯一的缺點就是嘴巴有時不太明顯。



除了單一類別描述以外，嘗試了同時含有多個類別的實驗，發現了些有趣的實驗，發現雖然訓練時是希望頭髮可以學到關於頭髮的資訊，但實際上除了頭髮資訊以外，各類別可能都還有隱含其他部位的資訊，如下圖，棕色頭髮在眼睛不為黃色和棕色時，容易把眼睛往藍色靠近；而將兩種頭髮顏色都設為 1 時，容易出現挑染的狀況，有時也會產生融合顏色的感覺，例如棕色和藍色會產生一部分像亞麻綠的顏色；而頭髮的顏色也會稍微影響眼睛的顏色的深淺。因此 ACGAN 中學到的類別，其實不一定完全獨立，會在人眼看不太出的地方稍作改變，也就是除了主要的標籤外，也會藏了一些其他部位的資訊。

III. Compare your model with WGAN, WGAN-GP, LSGAN

1. Model Description of the choosed model

我們選擇與 WGAN 比較

| Discriminator | Generator |
|--|---|
| Conv2d (filter = 64, kernel = 8) ReLU Avg_pooling(kernel=3, stride = 2) | Linear(16*16*128) SELU BatchNorm |
| Conv2d (filter = 64, kernel = 6) ReLU Avg_pooling(kernel=3, stride = 2) | Reshape ConvTranspose2d(128, 16) SELU |
| Conv2d (filter = 128, kernel = 4) ReLU Avg_pooling(kernel=3, stride = 2) | Conv2d(128, 4) SELU |
| Conv2d (filter = 256, kernel = 3) ReLU Avg_pooling(kernel=3, stride = 2) | ConvTranspose2d(64, 4) SELU BatchNorm |
| Linear(1024) ReLU Linear(1024) ReLU Linear(1) | Conv2d(3, 4) Tanh |
| learning rate = 2e-4 with RMSprop | learning rate = 2e-4 with RMSprop |
| Batch_size = 200 Lambda = 1 | |

基本上與我們做的 WGANGP 使用了幾乎完全一樣的參數，僅僅將 gradient penalty 轉換成 weight clipping。

2. Result of the model

使用 WGAN 產生的圖片如下：



| MS-SSIM | SWD-16 | SWD-32 | SWD-64 | SWD-Avg |
|---------|---------|---------|----------|---------|
| 0.4466 | 39.9442 | 30.9807 | 110.1012 | 60.3420 |

3. Comparison Analysis

我們的 WGANGP 與 WGAN 最大的差別在於，WGAN 會很容易出現毀容的狀況，這很明顯跟 WGANGP paper 中提到的問題，Weight 的分佈可能集中在 C 的區段內限縮了 model 的細節，而 c 的選擇無助於改善這個情況。

IV. Training tips for improvement

我們用同一個 model 測試了 Tips 中的 4、5、9。

4: BatchNorm

Batch Normalization :

在 Discriminator 中的每一層 Convolution 層的後面加入 batch_norm 層，每個 Batch 只放全部都是真實的資料或是全部都是生成的資料來訓練，從圖片上可以感覺到，雖然他的方格感還是蠻重的，但相較於其他的測試，產生出的圖片比較多元，從 MS-SSIM 中也可以看出

這個結果，對抗 mode collapse 的效果較好。



| MS-SSIM | SWD-16 | SWD-32 | SWD-64 | SWD-Avg |
|---------|---------|---------|----------|----------|
| 0.3011 | 48.2457 | 64.1113 | 221.8976 | 111.4182 |

5: Avoid Sparse Gradients: ReLU, MaxPool

Relu :

Generator 所有 Selu 改成 Relu

Relu 會比較容易出現某個特徵細節特別好，其他都很糟糕的狀況，有可能是只需要個別特徵就能夠支撐 discriminator 的判斷,但顯然不符合我們的期待，但從 SWD 的數值上會發現他產生的圖片與樣本空間最相像。



| MS-SSIM | SWD-16 | SWD-32 | SWD-64 | SWD-Avg |
|---------|---------|---------|---------|---------|
| 0.3701 | 23.5629 | 14.2108 | 67.6884 | 35.1540 |

Max pooling :

Discriminator Average pooling 改成 Max pooling

Max pooling 的特性就是將各個 filter 最大的值選出來, 和鄰近的 pixel 完全沒有任何交互作用 所以產生一堆挑染失敗的 8+9, 然後線條會比較明顯, 看起來比較有立體感, 比起一般 gan 水彩畫的風格, 比較像蜡筆畫, 眼睛、 臉型、 輪廓會比較明顯



| MS-SSIM | SWD-16 | SWD-32 | SWD-64 | SWD-Avg |
|---------|---------|---------|---------|---------|
| 0.4744 | 38.2372 | 24.1162 | 78.8118 | 47.0551 |

9: Use the ADAM Optimizer

Adam :

Generator 所有 RMSprop 改成 Adam , $\beta_1 = 0.5$, $\beta_2 = 0.5$

由於畫作必須要從整體來看像素之間的關係, 在多層 layer 下可能會上下的 neuron 來回滾過頭, 以至於 adam 比較難提高解析度或收斂, 很有可能細節會比較粗糙或是呈獻方格狀,



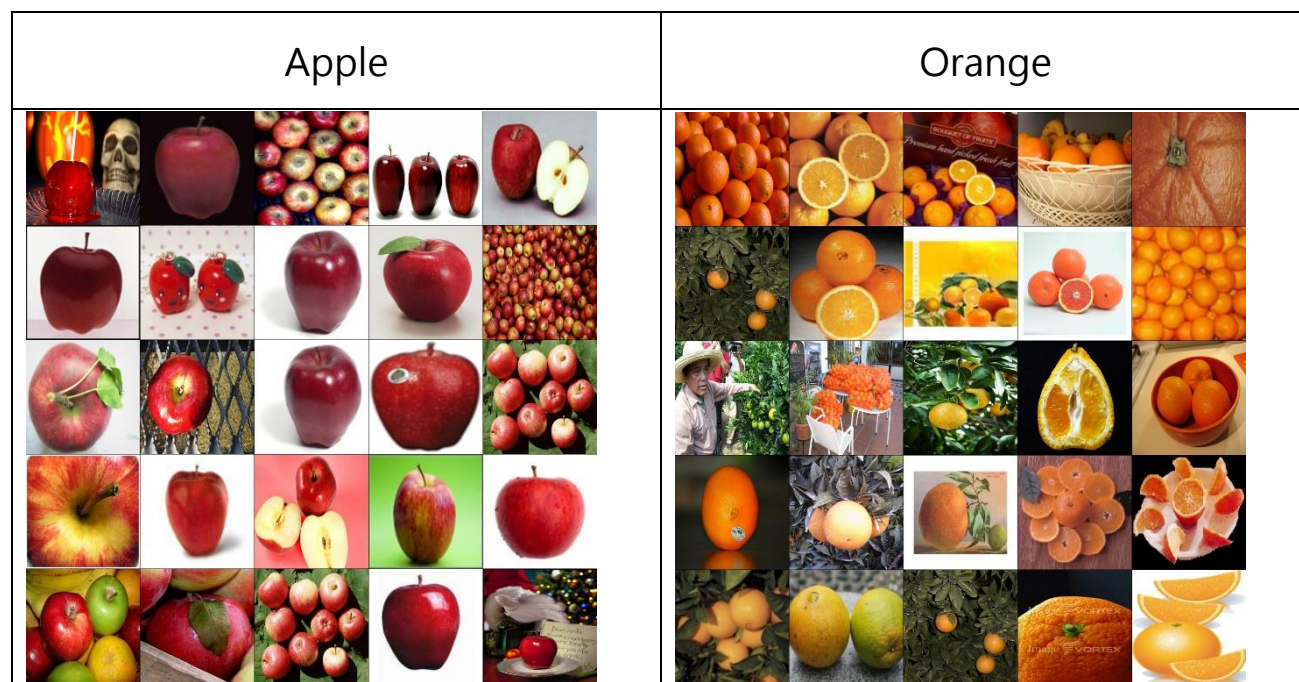
| MS-SSIM | SWD-16 | SWD-32 | SWD-64 | SWD-Avg |
|---------|---------|---------|---------|---------|
| 0.4214 | 40.2372 | 30.1162 | 88.8118 | 53.0551 |

V. Style Transfer

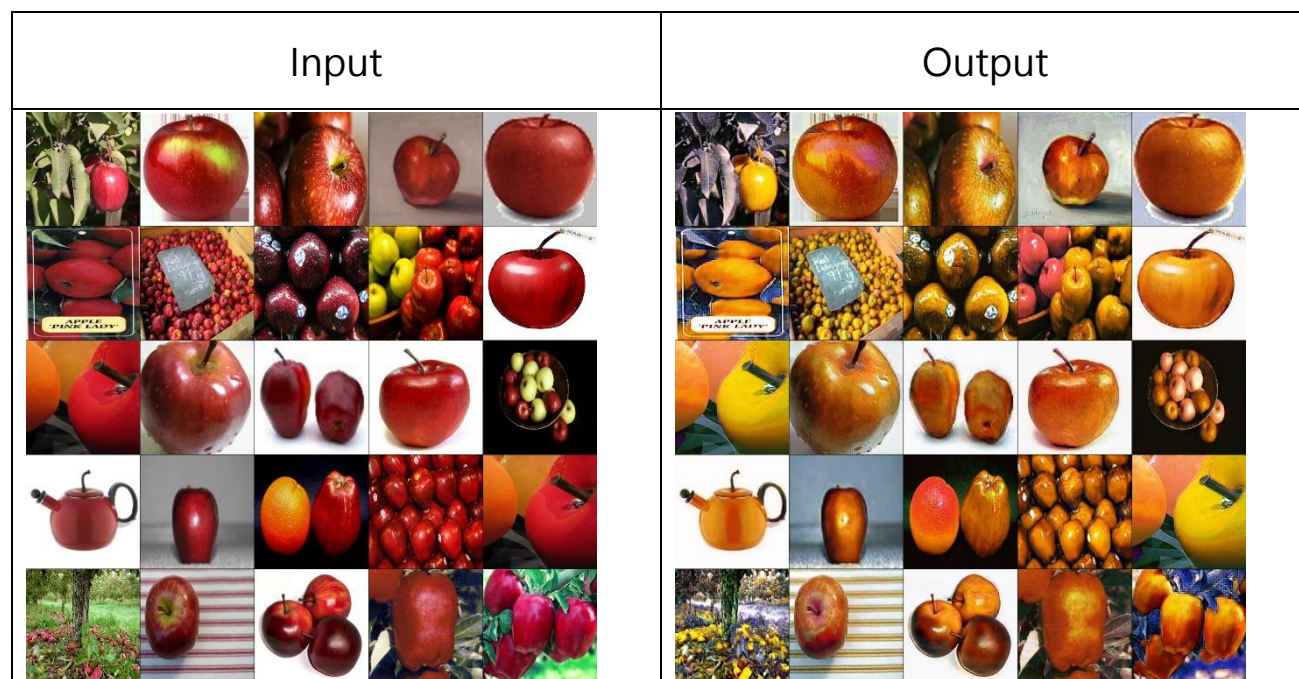
1. Show your result



我們使用 CycleGAN 的架構。測試的資料集為 apple2orange

原始的图片



Style transfer



| Input | Output |
|---|--|
|  |  |

2. Analysis

我們使用了 CycleGAN 的架構，使用了 junyanz 在 github[3]上 pre-trained 的 model。從這兩個資料集互相轉換的結果，可以發現這個模型學到的最主要是把顏色給改變，但是對於形狀的改變不大。所以會連不是水果部分的背景，也將顏色給改變。此外看起來像圓形的東西，較容易被認為是轉換目標，譬如蘋果轉成橘子的圖片中放了一個水壺，他會將整個水壺誤認是蘋果作風格轉換，但是對於邊緣沒有完整照出來的水果，轉換的效果看起來就比較差。整體來說這兩個圖片風格轉換看起來還是有蠻多破綻的，可能的原因是原作者用來訓練的資料集不大，訓練的資料 apple 共有 995 張，orange 共有 1019 張，或許如果有更多的資料集可以改善這個問題。

Reference:

[1] lbpcascade_animeface

https://github.com/nagadomi/lbpcascade_animeface

[2] Progressive Growing of GANs for Improved Quality, Stability, and Variation

<https://arxiv.org/abs/1710.10196>

[3] CycleGAN/vanhuyz

<https://github.com/vanhuyz/CycleGAN-TensorFlow>

分工表

| | |
|-----|----------------------|
| 蘇楷鈞 | 3-2 |
| 黃鈞宥 | 3-1、Tips 5、Tips 9 |
| 林羿辰 | 3-3、Tips 4、資料清理、評估方式 |