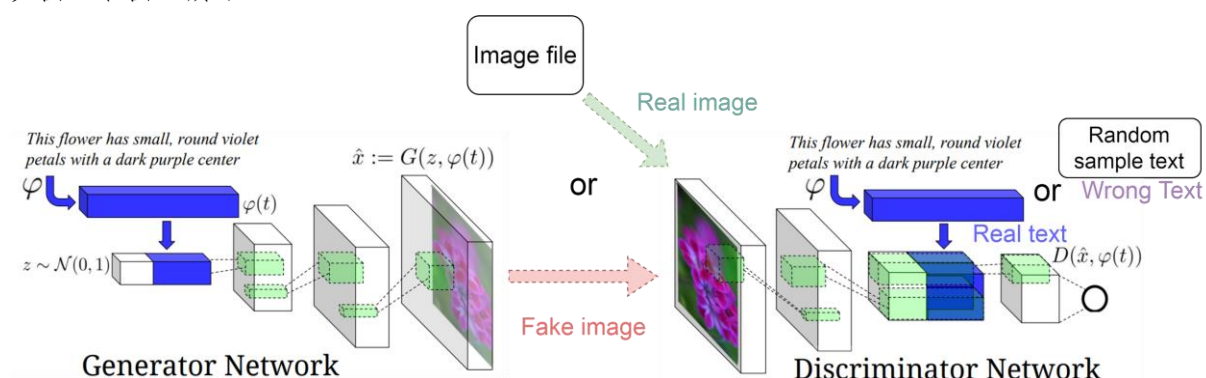


Preprocessing

./faces 中的圖片有 33431 張，然而有些 tag 不符合我們的需求，例如不包含髮色眼睛顏色的 tag，或者是只包含一種。因此我篩選那些同時包含 12 種髮色和 11 種眼睛顏色 tag 的圖片。這樣的圖有 11418 張。Condition 的部份我用 one-hot encoding 處理。每一種顏色搭配一個維度，所以 condition 是一個由 12 維 one hot 和 11 維 one hot 組成的向量。再將 condition 和 noise 給進模型時會經過 convolution，所以先將兩者的 shape 都弄成柱狀的，也就是 (23,1,1) 和 (100,1,1)，以方便操作。圖片的部分則是先轉成 64x64 再給進模型。

Model description

整體模型主要依照 Generative Adversarial Text to Image Synthesis 中提出的架構實作，如圖一。Generator 的部份需先將 100 維的 random noise 與 onehot vector 接起來，然後接著 upsampling 的操作。最後產生出 64x64x3 的 tensor 當作 fake image。Discriminator 則是給進圖片判斷真偽，因此有不同的 input 組合。分別為 (real img, right text)、(fake img, right text)、(real img, wrong text) 和 (wrong img, right text)，其中 (real img, right text) label 為 1，其他組合為 0。模型架構主要是由 CNN 組成，並且在最後一個 CNN 前將 text onehot vector 與 downsampling 後的圖片 tensor 接起來，一起過一個 CNN 後再 output。Generator 和 Discriminator 詳細的模型架構如圖二和圖三所示。



圖一、Model concept¹

```
generator (
  (deconv1): ConvTranspose2d(123, 512, kernel_size=(4, 4), stride=(1, 1))
  (deconv1_bn): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True)
  (deconv2): ConvTranspose2d(512, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
  (deconv2_bn): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True)
  (deconv3): ConvTranspose2d(256, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
  (deconv3_bn): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (deconv4): ConvTranspose2d(128, 64, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
  (deconv4_bn): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True)
  (deconv5): ConvTranspose2d(64, 3, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
)
```

圖二、Generator Network 架構

¹ <https://arxiv.org/pdf/1605.05396.pdf>

```

discriminator (
  (conv1): Conv2d(3, 64, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
  (conv2): Conv2d(64, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
  (conv2_bn): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (conv3): Conv2d(128, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
  (conv3_bn): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True)
  (conv4): Conv2d(256, 512, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
  (conv4_bn): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True)
  (conv5): Conv2d(512, 512, kernel_size=(4, 4), stride=(1, 1))
  (conv5_bn): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True)
  (linear): Linear (512 -> 1)
)

```

圖三、Discriminator Network 架構

訓練參數:

Updates between Generator and Discriminator: 2 : 1 or 3 : 1

ADAM with lr = 0.0002, momentum = 0.5

Gaussian noise dim = 100

batch size = 64

epoch = 250

objective function:

- Generator:

$$\min E_{h \sim p_h, z \sim p_z(z)} [-\log(D(G(z, h)))]$$

- Discriminator:

$$\min - \{ E_{x, h \sim p_{data}(x, h)} [\log D(x, h)] + E_{x \sim p_{data}(x, h), \hat{h} \sim p_{\hat{h}}, h \neq \hat{h}} [\log(1 - D(x, \hat{h}))] \\ + E_{h \sim p_{data}(x, h), \hat{x} \sim p_{\hat{x}}, x \neq \hat{x}} [\log(1 - D(\hat{x}, h))] + E_{h \sim p_h, z \sim p_z(z)} [\log(1 - D(G(z, h)))] \}$$

其中， $z = noise$, $x = real\ image$, $h = right\ text$, $\hat{x} = wrong\ image$, $\hat{h} = wrong\ text$

How do you improve your performance

1. 調整 Generator 和 Discriminator 的更新比例:

Generative Adversarial Networks 中文叫「生成對抗網路」，顧名思義就是兩個網路的抗衡。但不一定兩個網路都勢均力敵，所以藉由調整他們的更新比例來讓兩個網路的能力盡量相近。適當的調配比例可以讓頭髮眼睛顏色受控制。

2. 對圖像作擴增(flip, random crop, random rotate):

由於原本的訓練用圖片經篩選後剩 11418 張，容易使的訓練出來的圖片不夠清晰或者不受 condition 控制，因此藉由 data augmentation 的方法增加訓練資料。實作後可發現在適當的擴增下可以增加清晰程度。

3. WGAN²:

透過修改 objective function 和 Discriminator output，使得 GAN 的訓練更加穩定。實際的操作有:

- a. Discriminator 最後一層的 sigmoid 拿掉
- b. 對 Discriminator 做 weight clipping，限制在 [-c, c] 之間，本次作業取 0.01
- c. 使用 RMSprop 作為 optimizer (不過嘗試 adam 發現也能產生出還行的圖)

² <https://arxiv.org/pdf/1701.07875.pdf>

d. Loss function:

$$L_D^{WGAN} = E[D(x)] - E[D(G(z))]$$

$$L_G^{WGAN} = E[D(G(z))]$$

$$W_D \leftarrow \text{clip_by_value}(W_D, -0.01, 0.01)$$

4. WGAN with Gradient penalty (improved WGAN)³:

improved WGAN 的論文中提出一種替代 weight clipping 的方法，即懲罰 Discriminator 對輸入的梯度。實際操作如下：

a. 不做 weight clipping，並在 Discriminator loss 計算時加上 gradient penalty

b. Loss function:

$$L_D^{WGAN_GP} = L_D^{WGAN} + \lambda E[(|\nabla D(\alpha x - (1 - \alpha)G(z))| - 1)^2]$$

$$L_G^{WGAN_GP} = L_G^{WGAN}$$

Experiment settings and observations [Condition : red hair green eyes]

DCGAN:




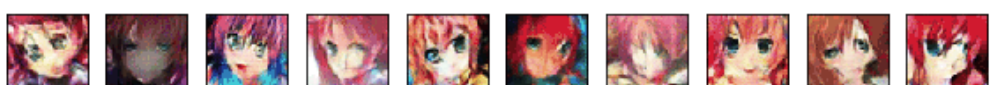

Setting A (basic) : (1)D and G Update ratio = 1:1 (2) No image augmentation

Setting B : (1)D and G Update ratio = 1:3 (2) No image augmentation

Setting C : (1)D and G Update ratio = 1:3 (2) Random crop (64) and flip (6 倍資料)

Setting D : (1)D and G Update ratio = 1:3 (2) 只 Flip (2 倍資料)

Setting E : (1)D and G Update ratio = 1:3 (2) Random crop (84) and flip (6 倍資料)

Setting A	
Setting B	
Setting C	
Setting D	
Setting E	

表一、conditional DCGAN 實驗

³ <https://arxiv.org/pdf/1704.00028.pdf>

WGAN:

雖然 paper 提到 WGAN 不需要小心平衡 Generator 和 Discriminator，但發現調整更新比例效果還是會有差異，所以也針對更新比例作了一些比較。

Setting A(basic) : (1)D and G Update ratio = 1:1 (2) image augmentation 只有 flip (兩倍資料)

Setting B: (1)D and G Update ratio = 1:2 (2) image augmentation 只有 flip (兩倍資料)

Setting C: (1)D and G Update ratio = 2:1 (2) image augmentation 只有 flip (兩倍資料)

Setting D: (1)D and G Update ratio = 3:2 (2) image augmentation 只有 flip (兩倍資料)

Setting E: (1)D and G Update ratio = 3:4 (2) image augmentation 只有 flip (兩倍資料)

Setting F: (1)D and G Update ratio = 5:1 (2) image augmentation random crop and flip (六倍)

Setting A	
Setting B	
Setting C	
Setting D	
Setting E	
Setting F	

表二、conditional WGAN 實驗

improved WGAN:

從 DCGAN、WGAN 的觀察中發現，若是 G 比 D 更新次數多圖會有點糊糊的。但增加 G 的更新次數可以提升顏色準確度。因此，improved WGAN 的實驗也是先從調整更新比例開始。設定如下：

Setting A(basic) : (1)D and G Update ratio = 1:1 (2) image augmentation 只有 flip (兩倍資料)





Setting B: (1)D and G Update ratio = 5:1 (2) image augmentation 只有 flip (兩倍資料)

Setting C: (1)D and G Update ratio = 5:2 (2) image augmentation 只有 flip (兩倍資料)

Setting D: (1)D and G Update ratio = 1:3 (2) image augmentation random crop and flip (六倍)

Setting E: (1)D and G Update ratio = 5:1 (2) image augmentation random crop and flip (六倍)

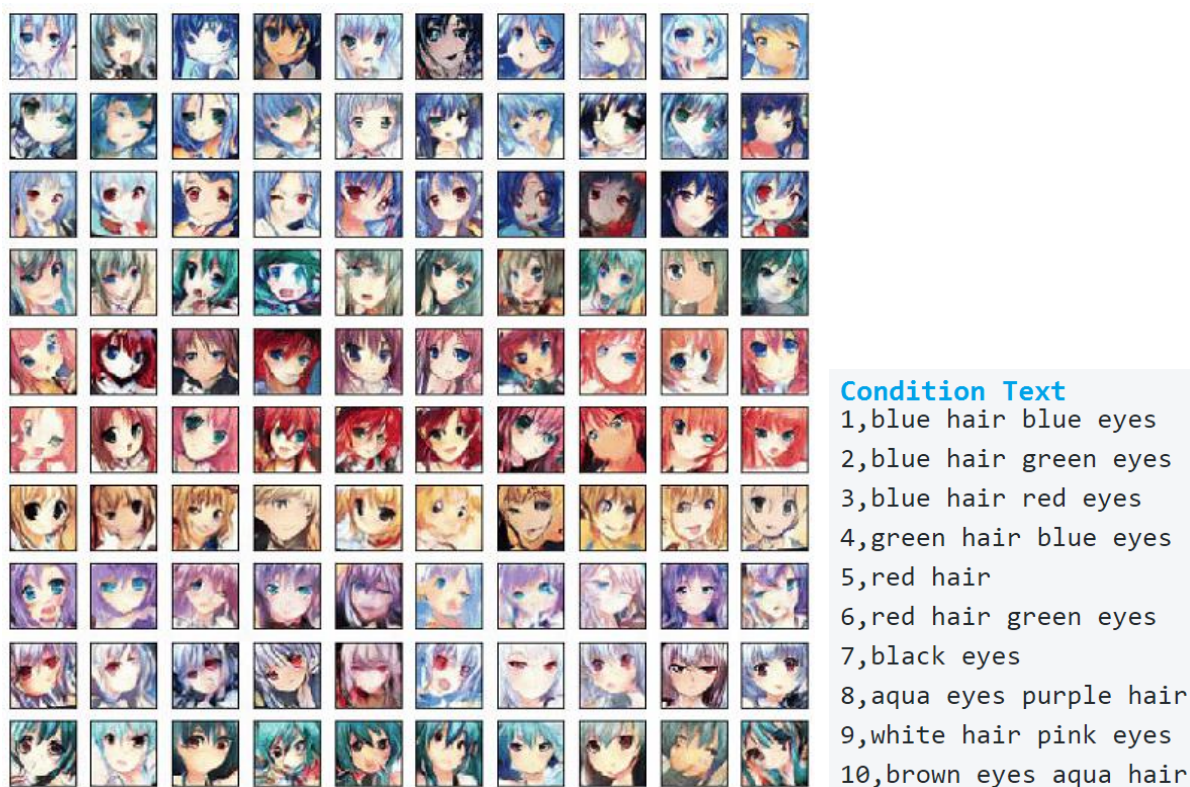
Setting A	
-----------	--

Setting B	
Setting C	
Setting D	
Setting E	

表三、conditional WGAN with gradient penalty 實驗

Comparison of three models:

從上述實驗結果看來，conditional DCGAN 就能發揮不錯的效果。其實 WGAN 的優勢並不是產生出來的圖會比 DCGAN 好，只是不需要花太多時間 fine tune 參數，平衡 Discriminator 和 Generator 等。若好好搭建 DCGAN 模型，效果是不錯的。而相較於 DCGAN，WGAN 和 improved WGAN 在顏色準確度上讓我花了不少時間琢磨。不過，相同資料量的情況下，我覺得 WGAN 產生出來的圖案比較好看，例如頭髮的輪廓比較細緻。為了讓 DCGAN 也能有更細緻的輪廓，我擴增六倍的資料量(random crop(88) -> resize(64), random rotate(5 degree) 和 flip)。實驗結果發現確實能提升清晰度，並且準確度也蠻高的。Best model 我使用 Discriminator : Generator = 1:3，6 倍資料的 augmented data，optimizer 使用 ADAM (lr = 0.0002, momentum = 0.5)。結果如下圖：



圖四、best result of conditional DCGAN