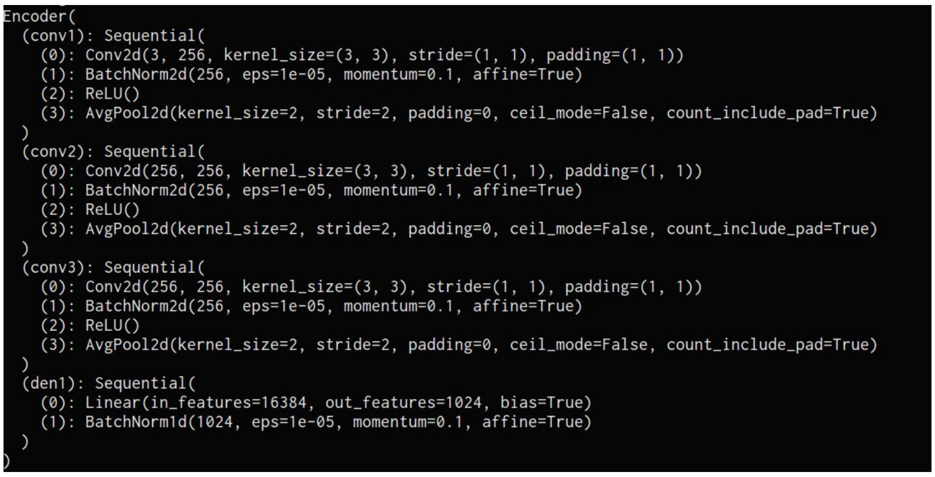
DLCV HW4 Report

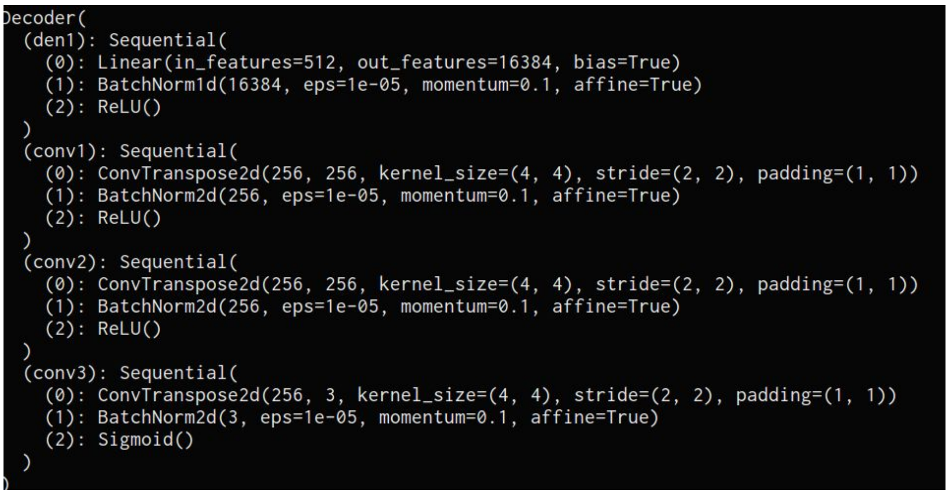
b04901136 張家銘
電機三

Problem 1. VAE

1. Describe the architectere & implementation details of your model.

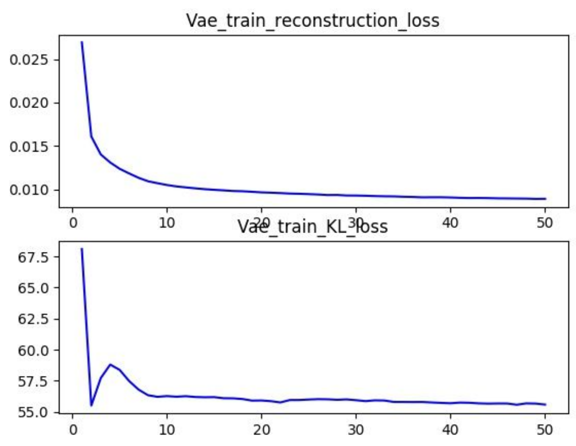


```
Encoder(
  (conv1): Sequential(
    (0): Conv2d(3, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True)
    (2): ReLU()
    (3): AvgPool2d(kernel_size=2, stride=2, padding=0, ceil_mode=False, count_include_pad=True)
  )
  (conv2): Sequential(
    (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True)
    (2): ReLU()
    (3): AvgPool2d(kernel_size=2, stride=2, padding=0, ceil_mode=False, count_include_pad=True)
  )
  (conv3): Sequential(
    (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True)
    (2): ReLU()
    (3): AvgPool2d(kernel_size=2, stride=2, padding=0, ceil_mode=False, count_include_pad=True)
  )
  (den1): Sequential(
    (0): Linear(in_features=16384, out_features=1024, bias=True)
    (1): BatchNorm1d(1024, eps=1e-05, momentum=0.1, affine=True)
  )
)
```



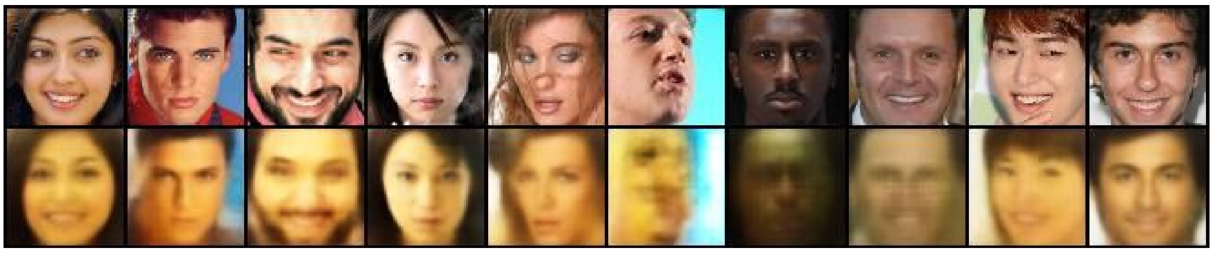
```
Decoder(
  (den1): Sequential(
    (0): Linear(in_features=512, out_features=16384, bias=True)
    (1): BatchNorm1d(16384, eps=1e-05, momentum=0.1, affine=True)
    (2): ReLU()
  )
  (conv1): Sequential(
    (0): ConvTranspose2d(256, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
    (1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True)
    (2): ReLU()
  )
  (conv2): Sequential(
    (0): ConvTranspose2d(256, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
    (1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True)
    (2): ReLU()
  )
  (conv3): Sequential(
    (0): ConvTranspose2d(256, 3, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
    (1): BatchNorm2d(3, eps=1e-05, momentum=0.1, affine=True)
    (2): Sigmoid()
  )
)
```

2. Plot the learning curve (reconstruction loss & KL divergence) of your model.



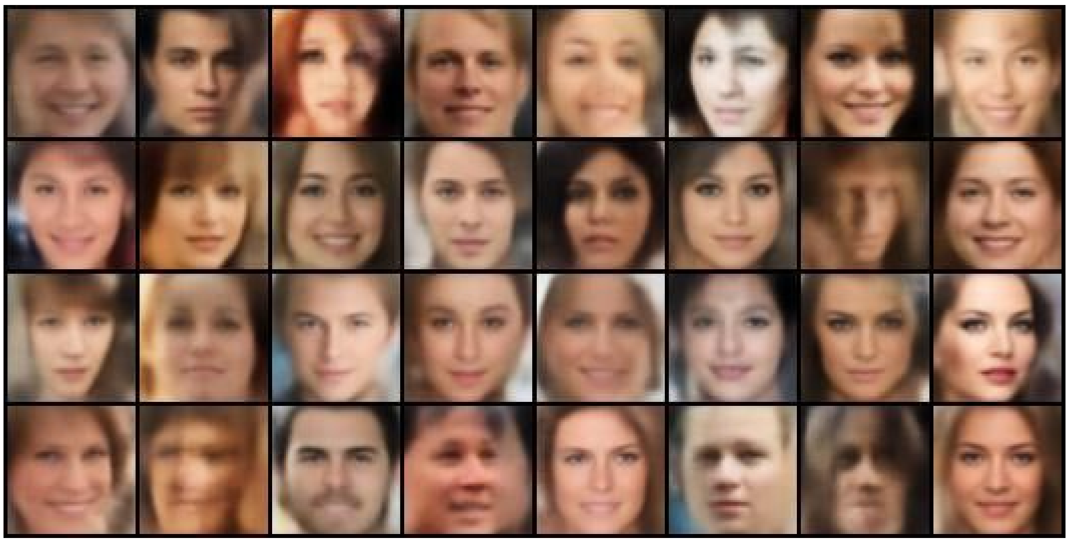
Reconstruction loss 是 decoder output data 跟原圖 normalize 後的MSE loss
KL loss 是 latent space 跟 Normal gaussian 的 KL divergance loss 除去 latent vector size.

3. Plot 10 testing images and their reconstructed result of your model and report your testing MSE of the entire test set.



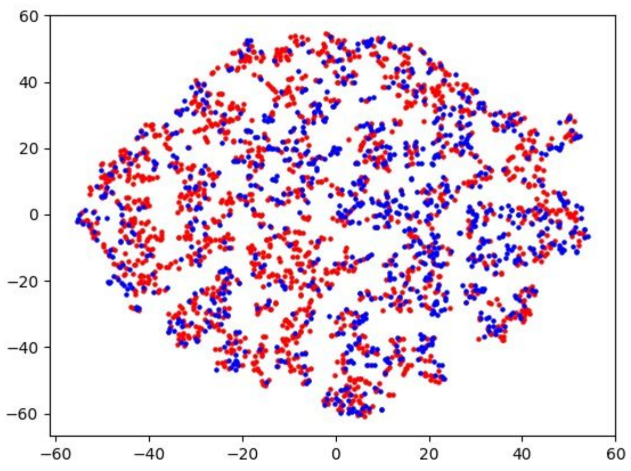
The final MSE loss on the entire test test is 0.01237

4. Plot 32 random generated images of your model.



5. Visualize the latent space by mapping test images to 2D space (with tSNE) and color them with respect to an attribute of your choice.

   I use Male as my feature.

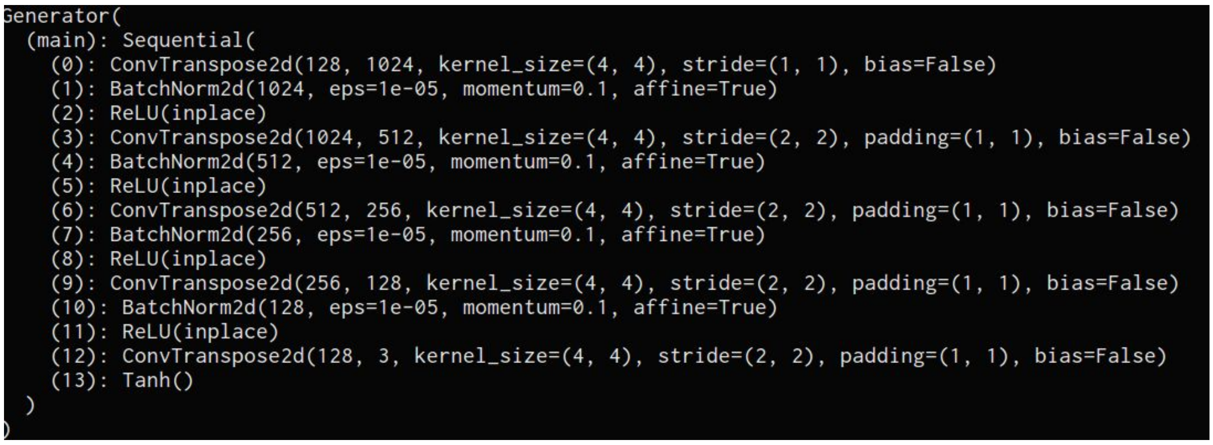


6. Discuss what you've observed and learned from implementing VAE.

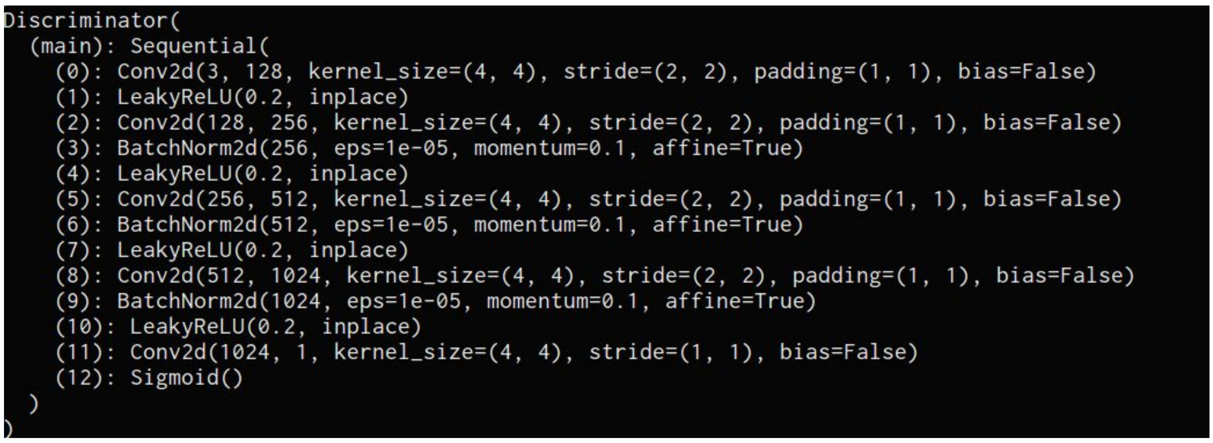
試了很多次參數後，我發現 1.KL Divergance coeffecient 很重要 2.參數如果調的很大，會全部圖形都差不多樣子， 所以我讓KL Divergance coefficient 在一開始的epo 中比較大，隨者epo愈小，會收斂到8E-5。

Problem 1. GAN

1. Describe the architectere & implementation details of your model.

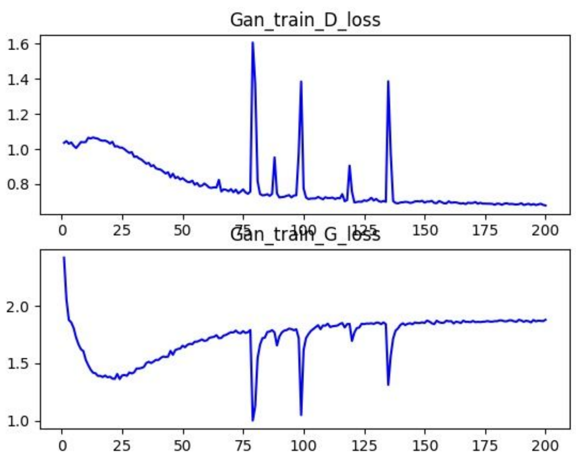


```
Generator(
  (main): Sequential(
    (0): ConvTranspose2d(128, 1024, kernel_size=(4, 4), stride=(1, 1), bias=False)
    (1): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True)
    (2): ReLU(inplace)
    (3): ConvTranspose2d(1024, 512, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (4): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True)
    (5): ReLU(inplace)
    (6): ConvTranspose2d(512, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (7): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True)
    (8): ReLU(inplace)
    (9): ConvTranspose2d(256, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (10): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
    (11): ReLU(inplace)
    (12): ConvTranspose2d(128, 3, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (13): Tanh()
  )
)
```



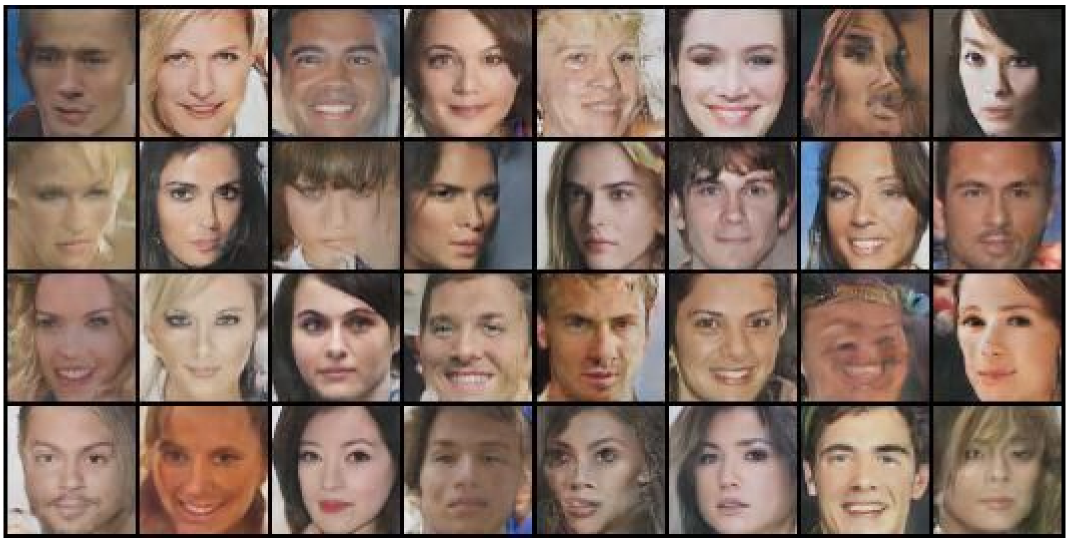
```
Discriminator(
  (main): Sequential(
    (0): Conv2d(3, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (1): LeakyReLU(0.2, inplace)
    (2): Conv2d(128, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (3): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True)
    (4): LeakyReLU(0.2, inplace)
    (5): Conv2d(256, 512, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (6): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True)
    (7): LeakyReLU(0.2, inplace)
    (8): Conv2d(512, 1024, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (9): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True)
    (10): LeakyReLU(0.2, inplace)
    (11): Conv2d(1024, 1, kernel_size=(4, 4), stride=(1, 1), bias=False)
    (12): Sigmoid()
  )
)
```

2. Plot the learning curve (in the way you prefer) of your model and briefly explain what you think it represents.



i. D loss 有 fake image 和 real image 的classification Binary CrossEntropy loss.
ii. G loss 只有 fake image 的classification Binary CrossEntropy loss.
iii. 很讓我驚訝的是，竟然有點convergance.

3. Plot 32 random generated images of your model.



4. Discuss what you’ve observed and learned from implementing GAN.

可能是training data 拍的角度都太不一樣，所以造成 data distribution 太分散，以至於有些 generated data 有些扭曲。

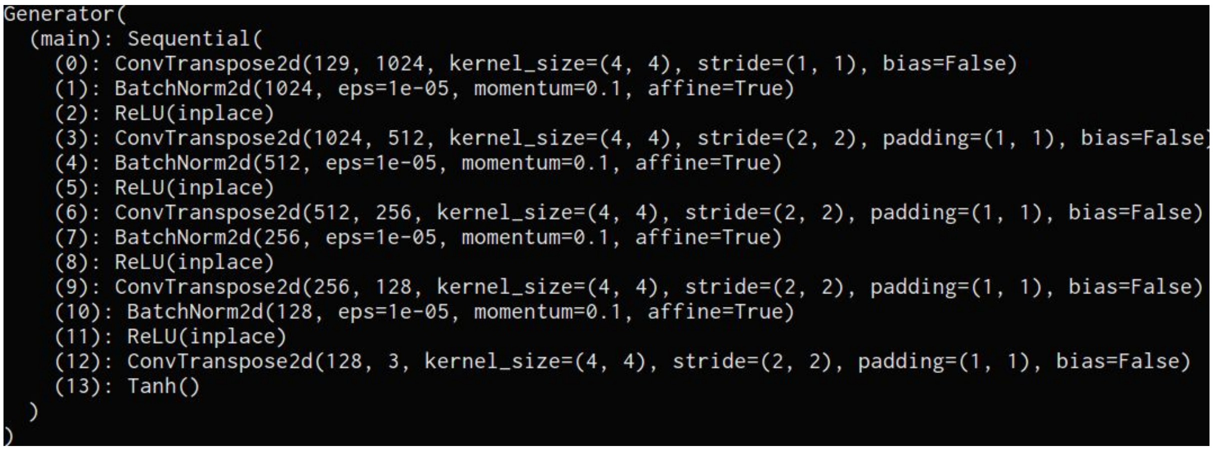
5. Compare the difference between image generated by VAE and GAN, discuss what you’ve observed.

VAE：照片不會有扭曲，甚至不合理的狀況產生，可是解析度都不高。

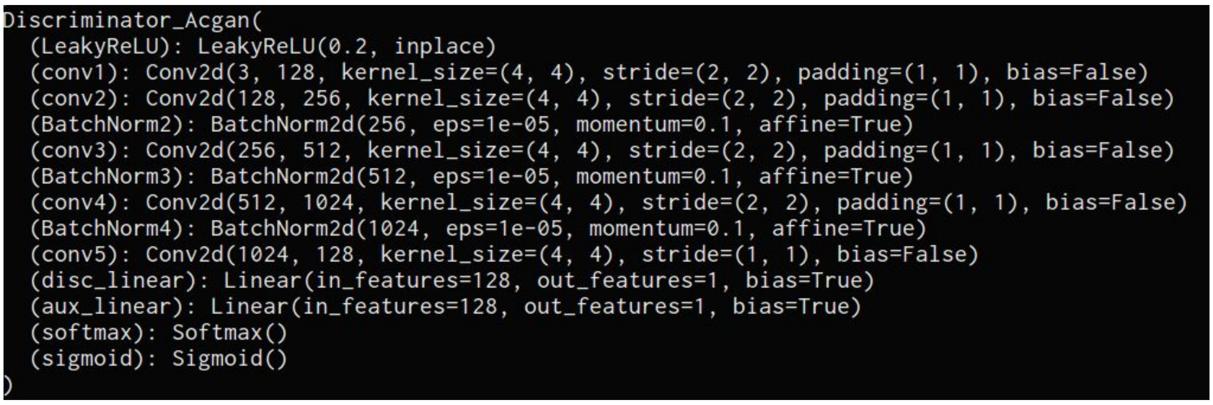
GAN：照片解析度高，可是有時候會有扭曲，甚至不合理的狀況。

Problem 1. ACGAN

1. Describe the architectere & implementation details of your model.

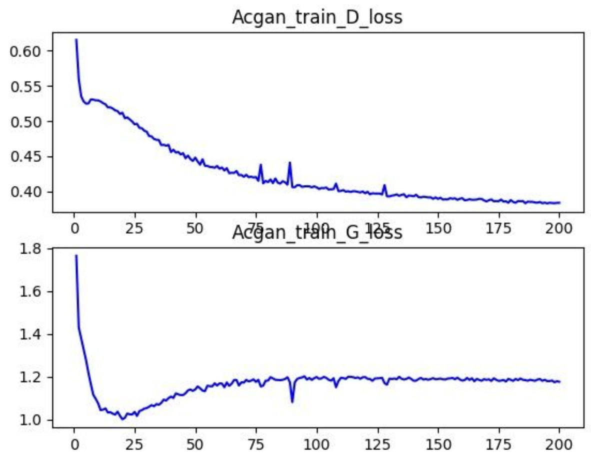


```
Generator(
  (main): Sequential(
    (0): ConvTranspose2d(129, 1024, kernel_size=(4, 4), stride=(1, 1), bias=False)
    (1): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True)
    (2): ReLU(inplace)
    (3): ConvTranspose2d(1024, 512, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (4): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True)
    (5): ReLU(inplace)
    (6): ConvTranspose2d(512, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (7): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True)
    (8): ReLU(inplace)
    (9): ConvTranspose2d(256, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (10): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
    (11): ReLU(inplace)
    (12): ConvTranspose2d(128, 3, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (13): Tanh()
  )
)
```



```
Discriminator_Acgan(
  (LeakyReLU): LeakyReLU(0.2, inplace)
  (conv1): Conv2d(3, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
  (conv2): Conv2d(128, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
  (BatchNorm2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True)
  (conv3): Conv2d(256, 512, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
  (BatchNorm3): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True)
  (conv4): Conv2d(512, 1024, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
  (BatchNorm4): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True)
  (conv5): Conv2d(1024, 128, kernel_size=(4, 4), stride=(1, 1), bias=False)
  (disc_linear): Linear(in_features=128, out_features=1, bias=True)
  (aux_linear): Linear(in_features=128, out_features=1, bias=True)
  (softmax): Softmax()
  (sigmoid): Sigmoid()
)
```

2. Plot the learning curve (in the way you prefer) of your model and briefly explain what you think it represents.



   i. D loss 有 fake image 和 real image 的 判斷真偽 Binary CrossEntropy loss, 以及 fake image 和 real image 的 判斷 class 的 Binary CrossEntropy loss.
   ii. G loss 有 fake image 的 判斷真偽 Binary CrossEntropy loss, 以及 fake image 的 判斷 class 的 Binary CrossEntropy loss.
   iii. 很讓我驚訝的是，竟然有點convergance.

3. Plot 10 pair of random generated images of your model, each pair generated from the same random vector input but with different attribute. This is to demonstrate your model's ability to disentangle feature of interest.