DLP LAB7

309553008

- Introduction

  利用兩種 Generator model(GAN, NF)生成指定條件的圖片

  Training data 為 ICLEVR 的幾何圖片，有 3 種形狀 8 種顏色，共 24 種不同物體，最終的要求是從某個 latent space output 出接近 really image 的圖片，評分(像不像 really image)由 TA 給的 model 來評

- Implementation details

  我使用 cDCGAN 的架構

  GAN 的部分:

  - Hyperparameters:

```
z_dim = 100
c_dim = 64*64
image_shape = (64,64,3)
epochs = 200
lr = 0.0002
batch_size = 64
```

  - Generator:

```
def forward(self, z, c):
    z = z.view(-1, self.z_dim, 1, 1)
    c = self.condition_resize(c).view(-1, self.c_dim, 1, 1)
    output = torch.cat((z, c), dim=1)
    output = self.conv1(output)
    output = self.conv2(output)
    output = self.conv3(output)
    output = self.conv4(output)
    output = self.conv5(output)
    output = self.tanh(output)
```

```
class Generator(nn.Module):
    def __init__(self, z_dim, c_dim):
        super(Generator, self).__init__()

        self.z_dim = z_dim
        self.c_dim = c_dim
        self.condition_resize = nn.Sequential(
            nn.Linear(24, c_dim),
            nn.ReLU()
        )

        channels = [z_dim+c_dim,512,256,128,64]
        paddings=[(0,0),(1,1),(1,1),(1,1)]
        for i in range(1,len(channels)):
            setattr(self,'conv'+str(i),nn.Sequential(
                nn.ConvTranspose2d(channels[i-1], channels[i], kernel_size=(4, 4), stride=(2, 2), padding=paddings[i-1]),
                nn.BatchNorm2d(channels[i]),
                nn.ReLU()
            ))
        self.conv5 = nn.ConvTranspose2d(64, 3, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
```

先將 condition 經過 FC 擴大成 64*64 維，再將 latent z 與擴大的 condition cat 再一起，最後經過 5 個 ConvTranspose output 64*64*3 的 fake image

- Discriminator

```
def forward(self, x, c):
    #   make condition be (N, 1, 64, 64)
    c = self.condition_resize(c.float()).view(-1, 1, self.H, self.W)
    #   (N, 4, 64, 64)
    output = torch.cat((x, c), dim=1)
    output = self.conv1(output)
    output = self.conv2(output)
    output = self.conv3(output)
    output = self.conv4(output)
    #   (N, 1, 1, 1)
    output = self.conv5(output)
    output = self.sigmoid(output)
    #   true / false
    output = output.view(-1)

    return output
```

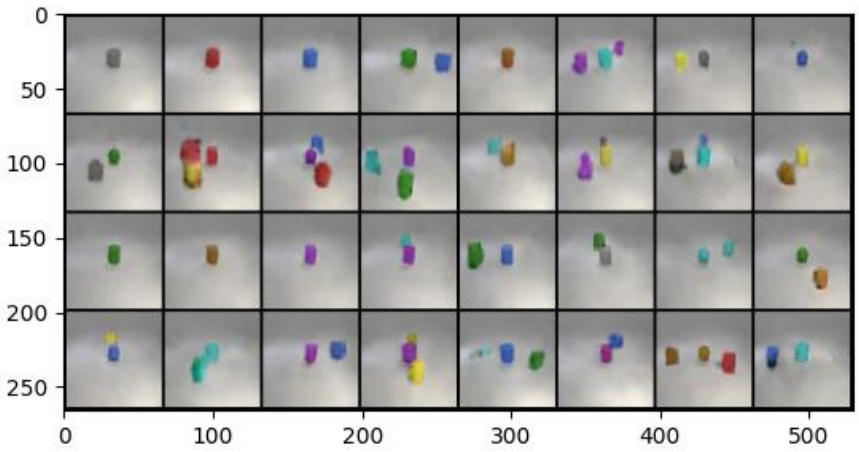
```
class Discriminator(nn.Module):
    def __init__(self, img_shape):
        super(Discriminator, self).__init__()

        self.H, self.W, self.C = img_shape
        self.condition_resize = nn.Sequential(
            nn.Linear(24, self.H*self.W*1),
            nn.ReLU()
        )

        channels = [4,64,128,256,512]
        for i in range(1, len(channels)):
            setattr(self, 'conv'+str(i), nn.Sequential(
                nn.Conv2d(channels[i-1], channels[i], kernel_size=(4, 4), stride=(2, 2),padding=(1, 1)),
                nn.BatchNorm2d(channels[i]),
                nn.LeakyReLU()
            ))
        self.conv5 = nn.Conv2d(512, 1, kernel_size=(4, 4))
        self.sigmoid = nn.Sigmoid()
```

將 condition 經過 FC 擴大成 64*64 維後與 input x cat 再一起，再經過 5 個 Convolution 拉成(1, 1, 1)，最後經由 sigmoid 把數值壓到[0, 1] (true / false)

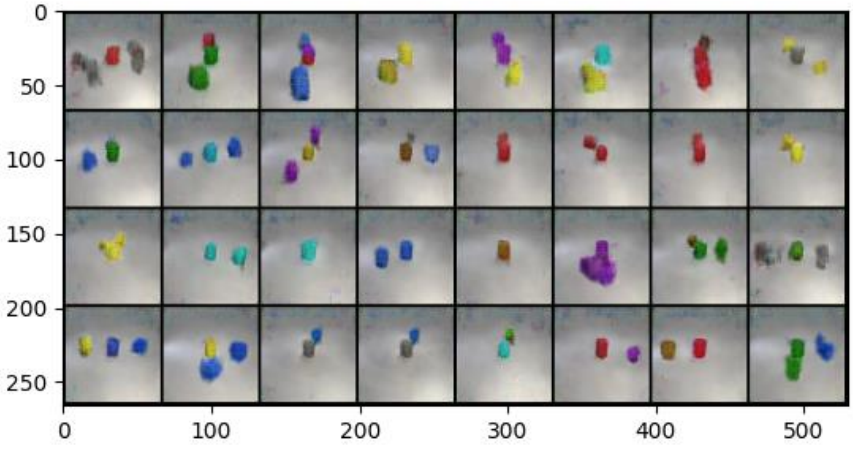
由於是 2 元分類問題，所以 loss function 採用 binary cross entropy

- Results and discussion
  - Task 1
  - GAN

Test: `score: 0.6388888888888888`



New test: `score: 0.5476190476190477`



我試過把架構改成 cWGAN，不過不知道是寫錯還是怎樣 train 不起來，score 都在 0.4 以下

- NF:
- 寫不出來…