Data preprocess:

這次功課的前處理我先將binary檔裡的資料使用data\_prerocess.py, data\_prerocess2.py，將img, label轉成npy檔。因為我發現如果直接在dataloader裡取用binary檔的資料會使cpu在檔案讀取方面拖太久的時間，

下圖是兩者的比較(上未轉npy，下轉npy)可以發現兩者在讀取64\*16筆資料時時間相差至13倍。

一張含有 文字 的圖片

自動產生的描述

兩種dataloader比較圖 Bonus dataset 是有\_2的檔案



一張含有 文字, 填字遊戲 的圖片

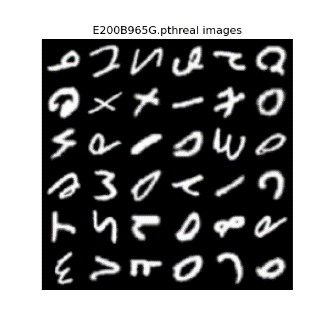
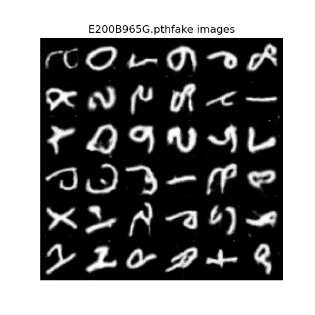
自動產生的描述

65 classes dataset Bonus 11 classes dataset

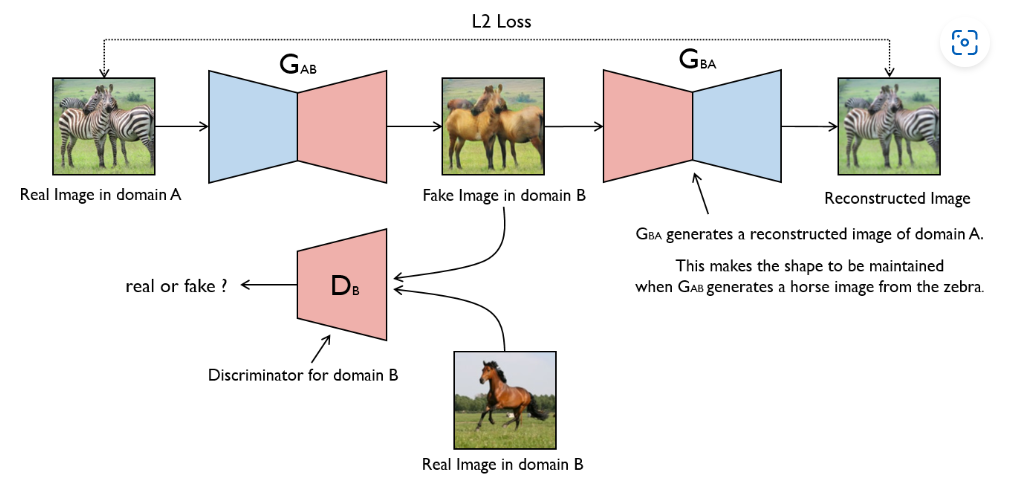
Generate model choise:

在generate model 方面我選擇的3個GAN分別為:

1. DCGAN (助教給的範例)
2. CycleGAN (聽說cycleGAN就能生產出很好的結果)
3. Conditional Diffusion model (另外找到的輕量化diffusion model)
4. DCGAN

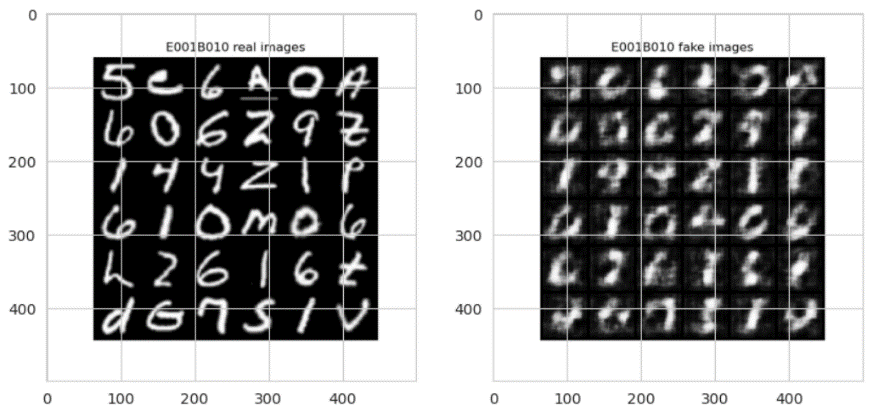
dcGAN的話就是直接沿用助教的設定下去訓練，結果如下二圖。

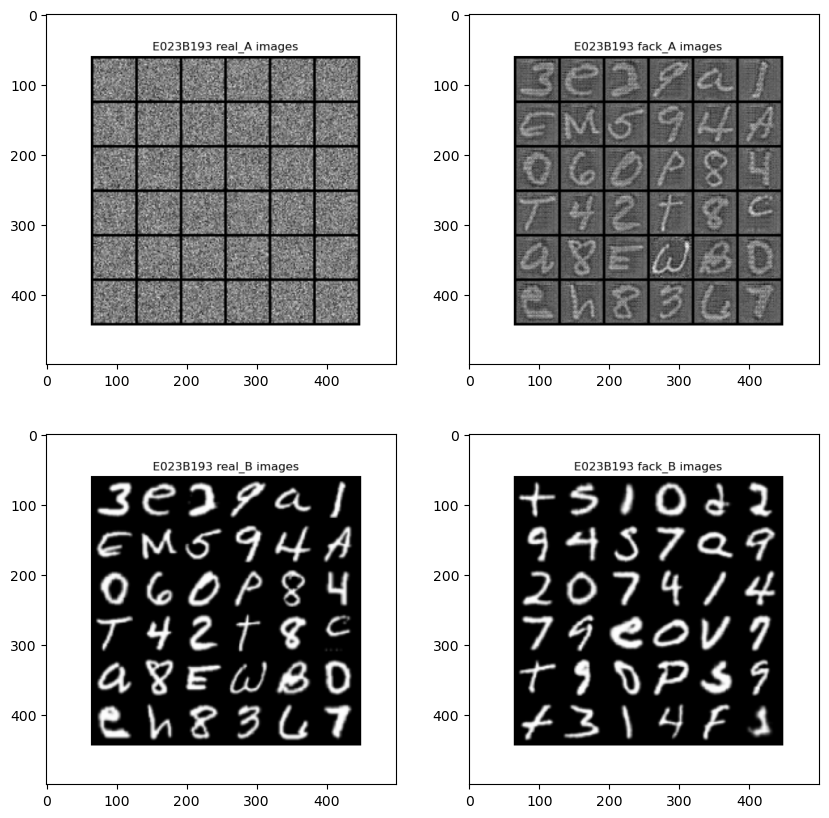
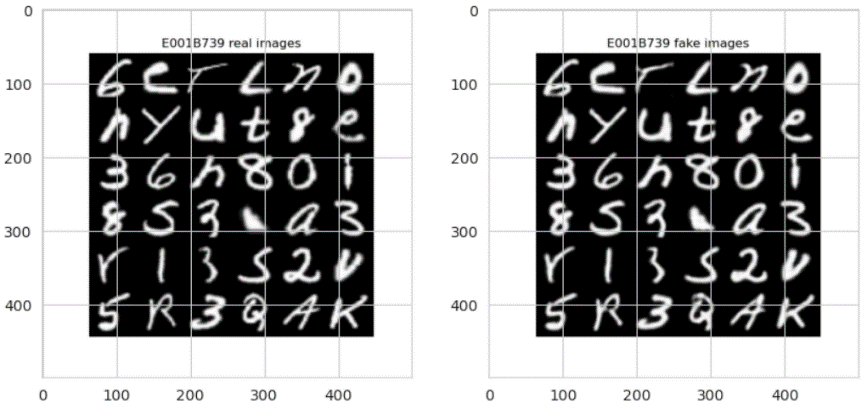
1. CycleGAN

因為cycleGAN都是IMG2IMG，風格轉換上的任務，像是馬轉斑馬，

通常需要一張real A圖，目標是轉成real B圖，這樣的前提下我以兩種想法下去做訓練:

1. Real A: a random sample imgs, Real B: shuffle Real A imgs
   1. Train 2個epoch就已經有很好的結果了，結果比較傾向autoencoder
2. Real A: a noise sample, Real B: a random sample imgs
   1. 這個模式train比較慢，大概50個epoch可以有不錯的結果

 I II



1. Conditional Diffusion model

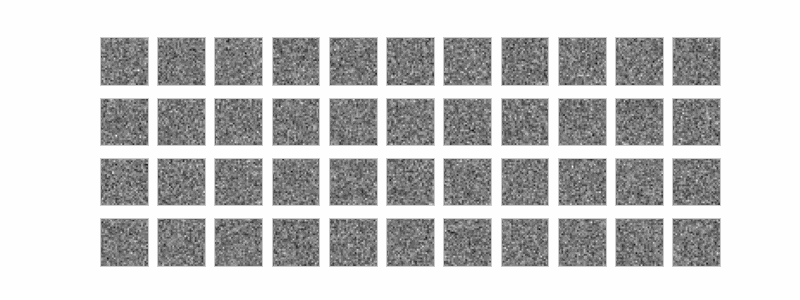
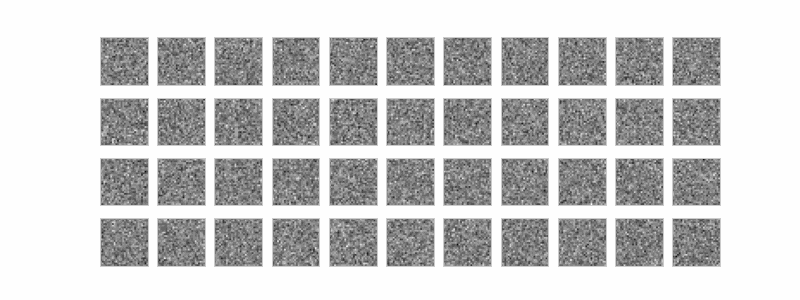
這是一個加了conditional 的 stable diffusion model，其中n\_T代表diffusion總共的步數，classes為Bonus的dataset數量，在訓練時我們input 的label會先經過enbeding 再跟圖片一起做訓練。 右圖是difussion model schedule的implement。最後我們也是照著這個schedule去跟generate出來的圖片去做L2 loss。

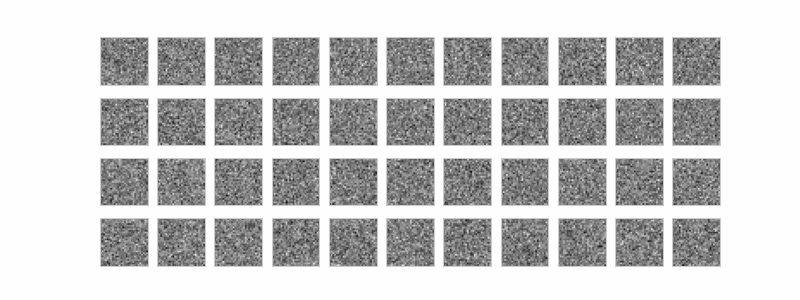
一張含有 文字 的圖片

自動產生的描述一張含有 文字 的圖片

自動產生的描述

從W\_0~W\_2代表的是context mask的程度多寡(conditional 的部分)，W\_0在inference的時候完全mask context，W越大代表給較多context的資訊。產生的圖也就越好。



W\_0 W\_1

一張含有 文字, 填字遊戲 的圖片

自動產生的描述

W\_2

實際上在inference的時候我們給定label，跟一個nomal nosie，model負責幫我們生產出跟label相關的圖片。(最後的W\_0~2是gif，想看的我都放在原始的word檔裡了)