1.

1.1：

1.

(a)我會選擇 spaCy，因為 spaCy 相較於 NLTK 效率較高，且 spaCy 十分有名，網路上有許多社群負責維護。(b)可以使用 whitespace 去 token，甚至有個 tokenizer 就叫 whitespace tokenizor.(c)複雜分詞器是因為句子中會有許多符號，需要做處理。

這裡使用 NLTK 和 spaCy 舉例，同樣都是一段文字，但因為不同 tokenizer 會 token 出不同長度的 output。

"""Founded in 2002, SpaceX’s mission is to enable humans to become a spacefaring civilization and a multi-planet species by building a self-sustaining city on Mars. In 2008, SpaceX’s Falcon 1 became the first privately developed liquid-fuel launch vehicle to orbit the Earth."""

NLTK 是 50，torchText 是 56

2.

special tokens ，⟨pad⟩是因為每句句子長度不會都相同，需要做填充，⟨unk⟩是因為要替換不適合您訓練詞彙的稀有詞。

3.

我是使用 spacy 作為 tokenizer，我並沒有先設定我 max_seq 的長度，而是寫出能自動抓取 train_data 中最常的句子長度，之後再+10 以防 test_data 超過長度。token 完之後再使用 torchtext 的 vocab 去作出 vocabulary。我原本使用 spacy 是想說他有許多功能像是 entity 之類的可以使 accuracy 提高，但在 code 裡我沒有用到。

1.2

1.

已經 pass baseline

2.

Attention map

3.

我的超參數為

Train_batchz_size = 100，我覺得一次 load 多一點數據他泛用性會更好

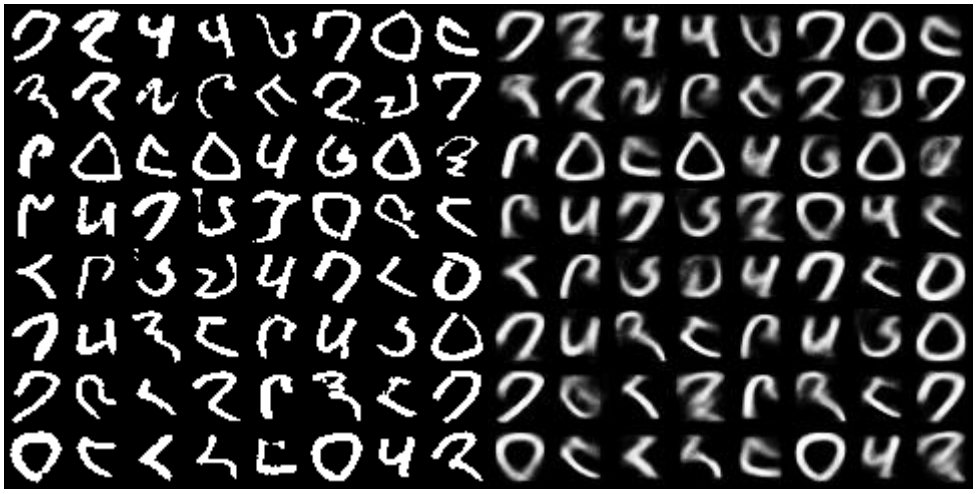
Num_hid = 100，網路我沒有設的很深

Num_head = 10，這個的要求要能整除 embed_dim(200)，因為 embedding layer 是 glove 6b200d

Num_layers = 2，我的電腦不允許我設太深
Dropout = 0.5，能有效防止 over fitting
Epochs= 設那麼多是因為我的 learning rate 最後會變很小，我怕還沒訓練完 epochs 就沒了
LR = 1e-4，使用 consinewarmupscheduler，lr 隨 epoch 改變
Clip_grad =1，normalization 使用

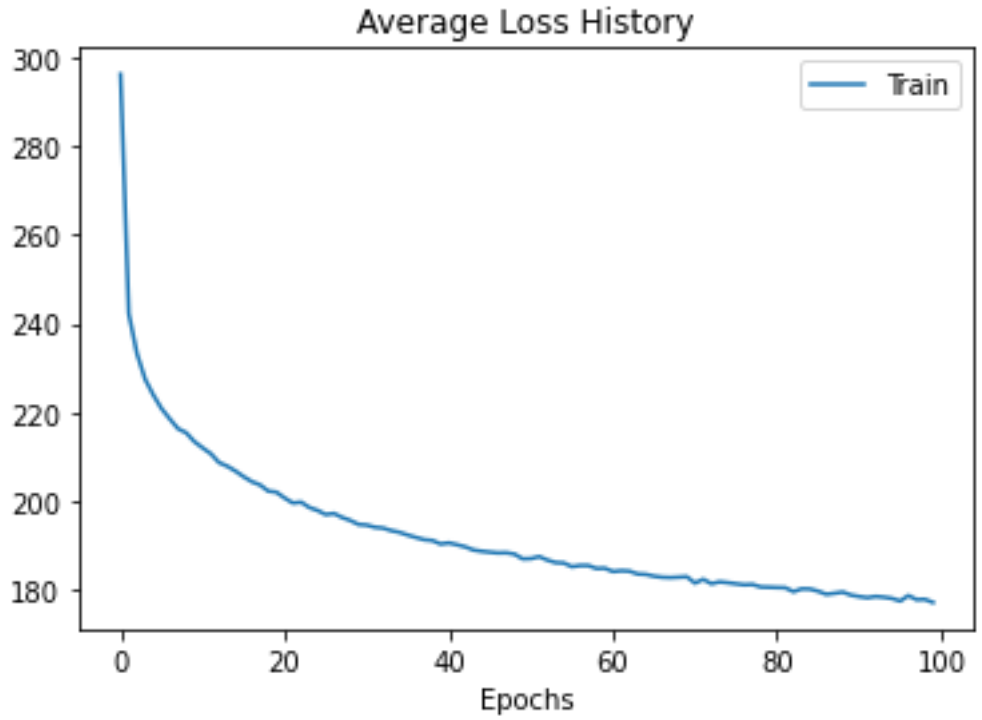
2.

1.首先是 TibetanMNIST
Lambda = 1 時
TibetanMNIST 的 real samples(左) 和 reconstructed samples(右)



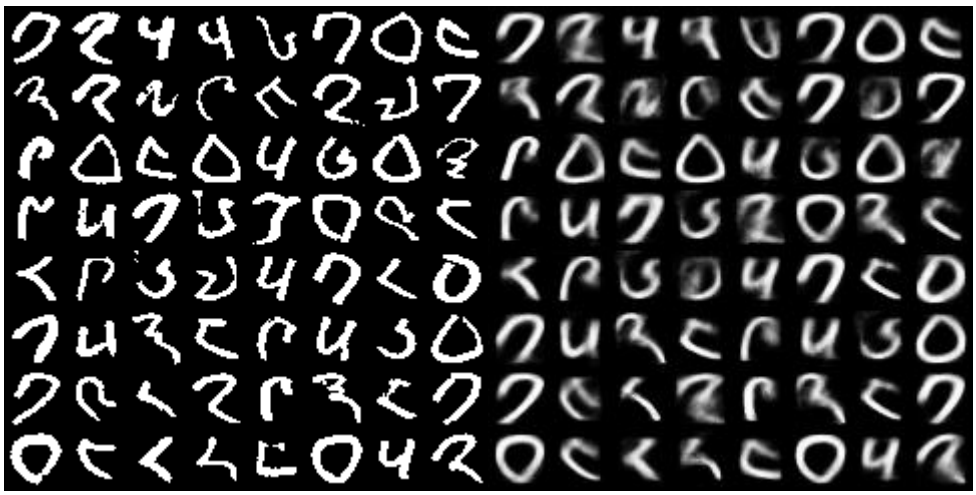
TibetanMNIST 的 fake(左) 和 interpolation(右)

TibetanMNIST 的 loss 曲線



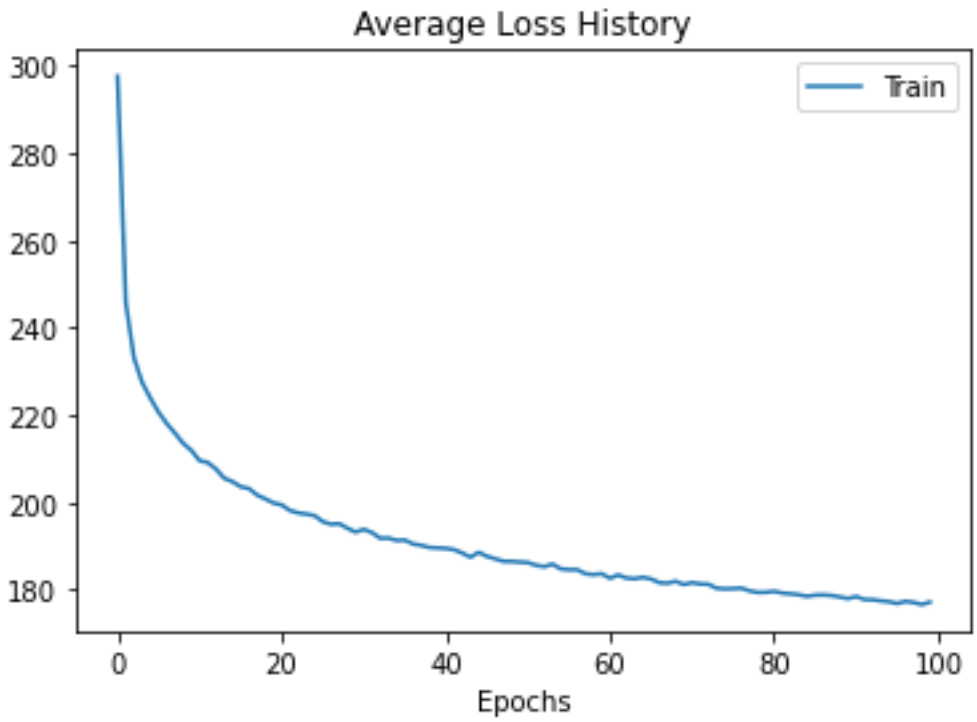
Lambda = 0 時

TibetanMNIST 的 real samples(左) 和 reconstructed samples(右)



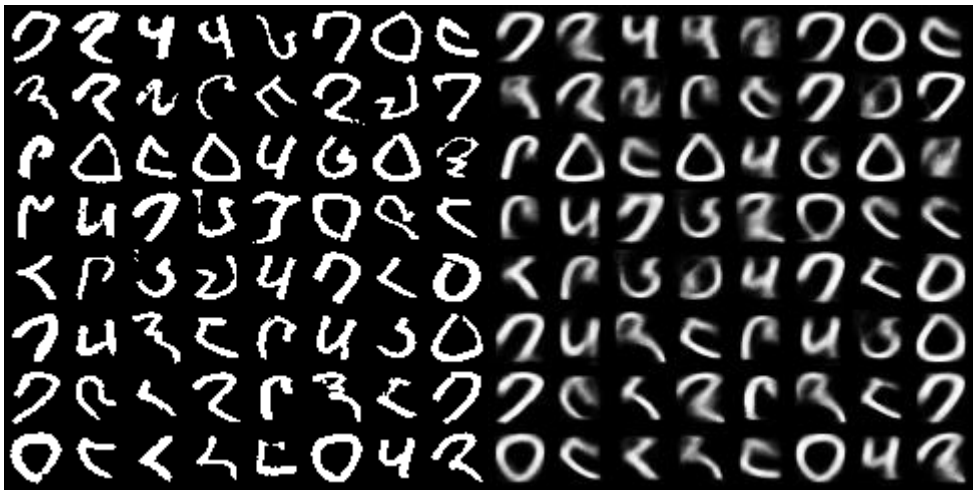
TibetanMNIST 的 fake(左) 和 interpolation(右)

TibetanMNIST 的 loss 曲線



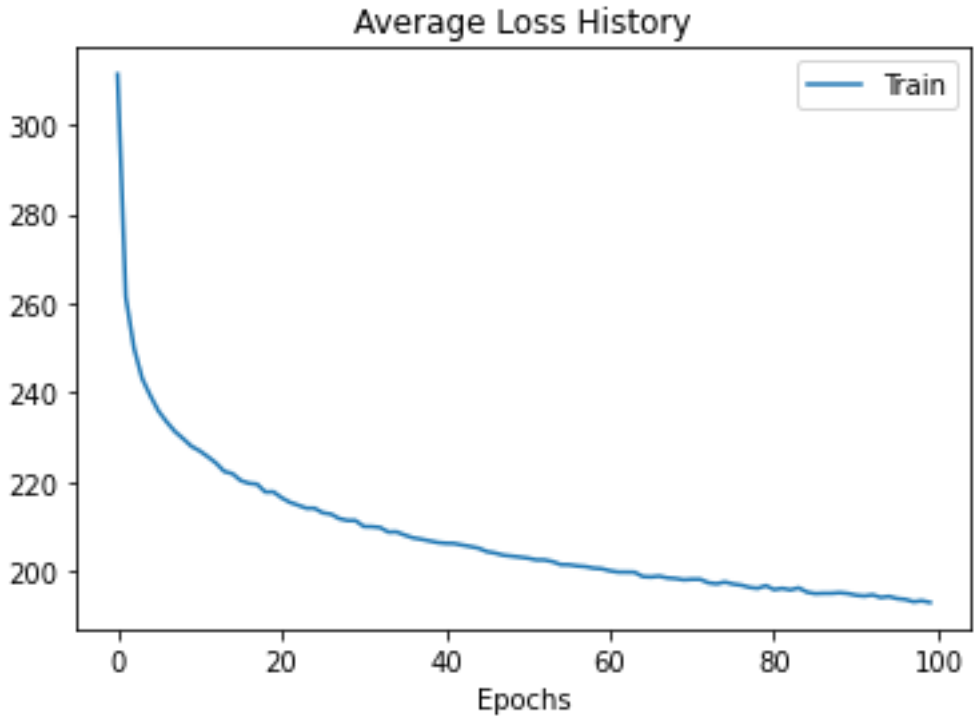
Lambda = 100 時

TibetanMNIST 的 real samples(左) 和 reconstructed samples(右)



TibetanMNIST 的 fake(左) 和 interpolation(右)

TibetanMNIST 的 loss 曲線

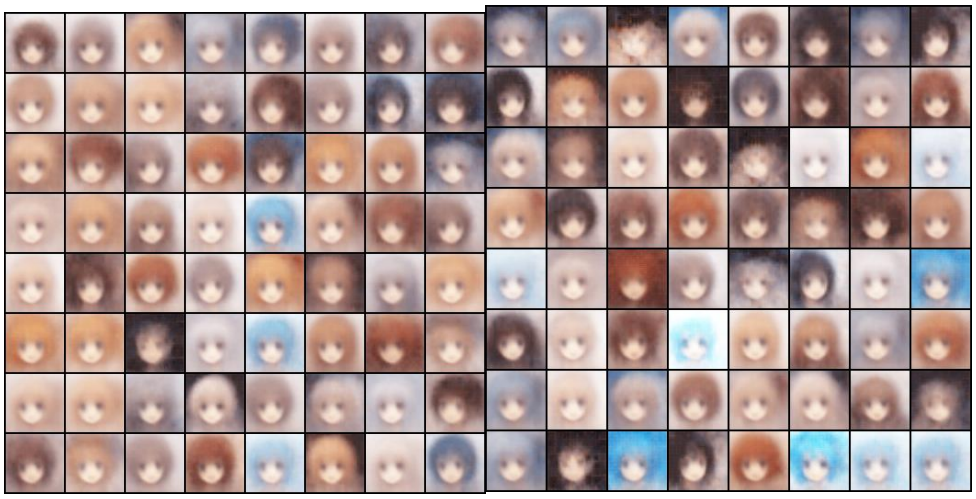


2.接著是 Amine_faces

先放上原本 real samples

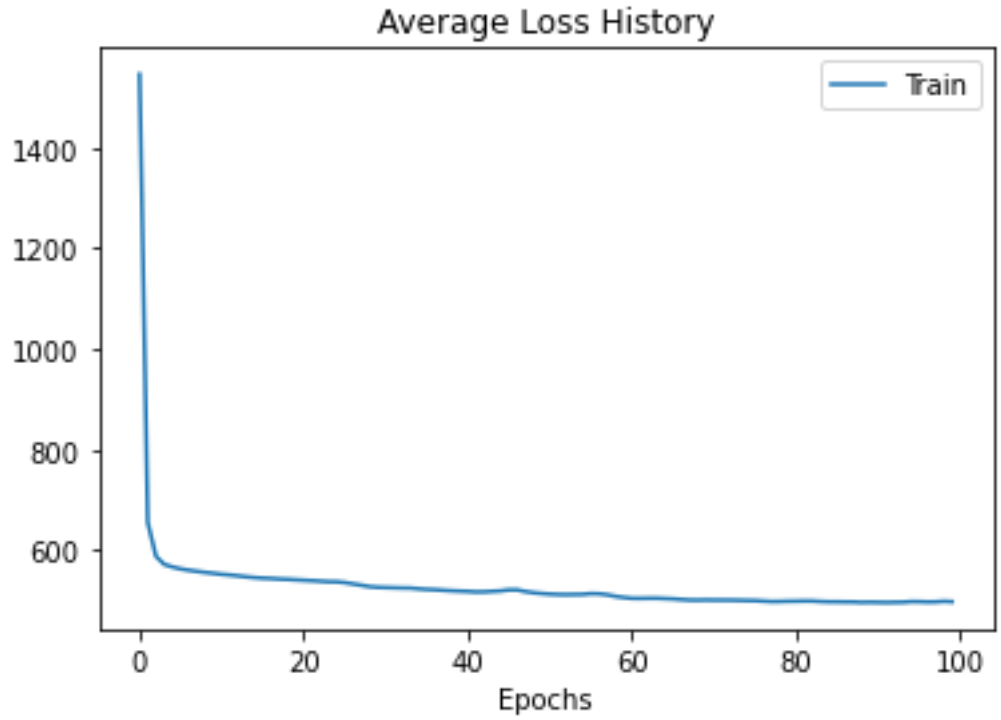
在 Lambda = 1 時

Amine_faces 的 reconstructed samples(左) 和 fake(右)



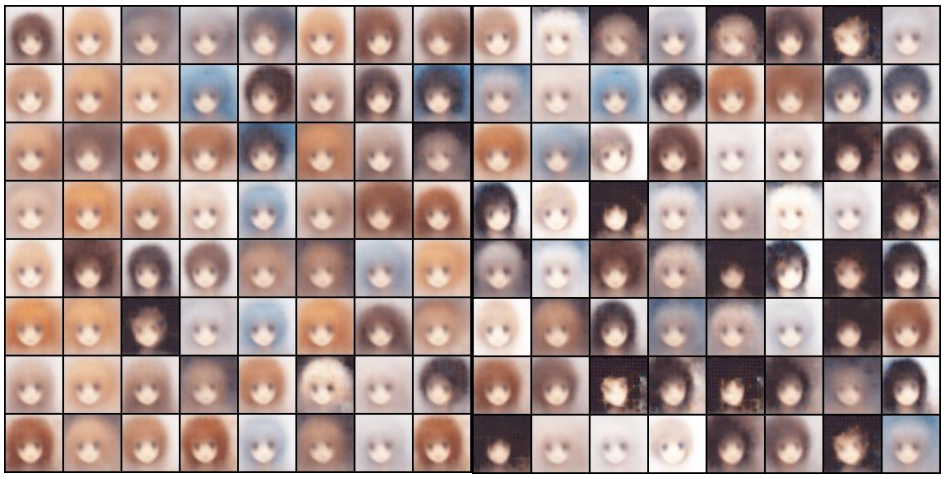
Amine_faces 的 interpolation

Amine_faces 的 loss curve



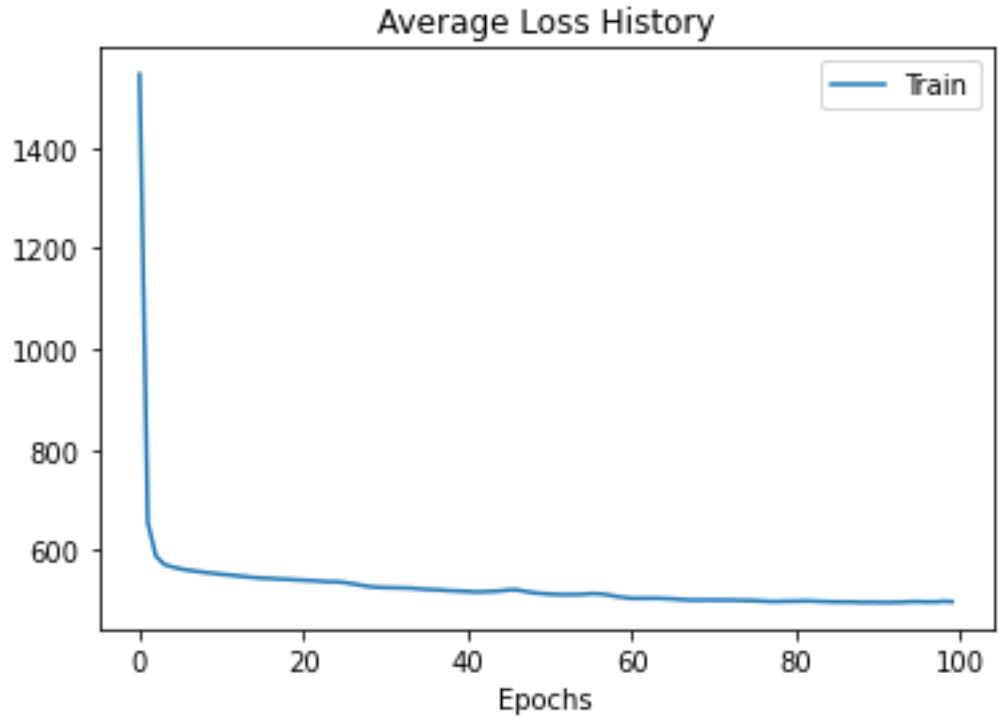
在 Lambda = 0 時

Amine_faces 的 reconstructed samples(左) 和 fake(右)



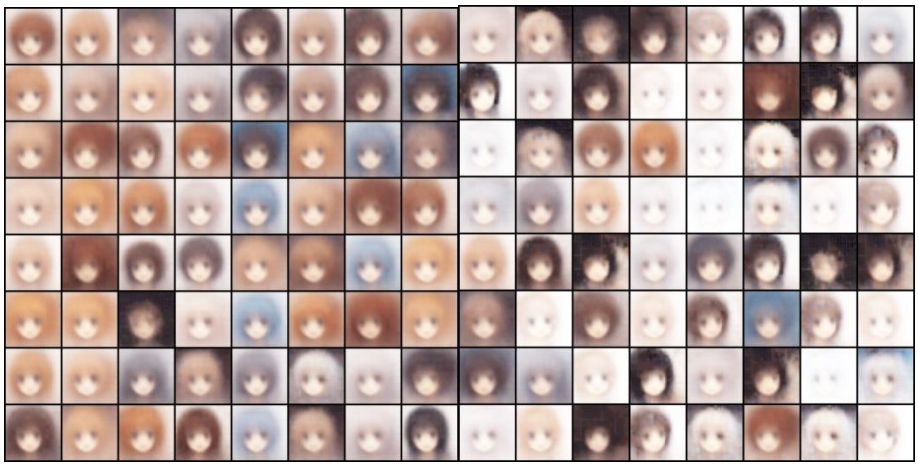
Amine_faces 的 interpolation

Amine_faces 的 loss curve



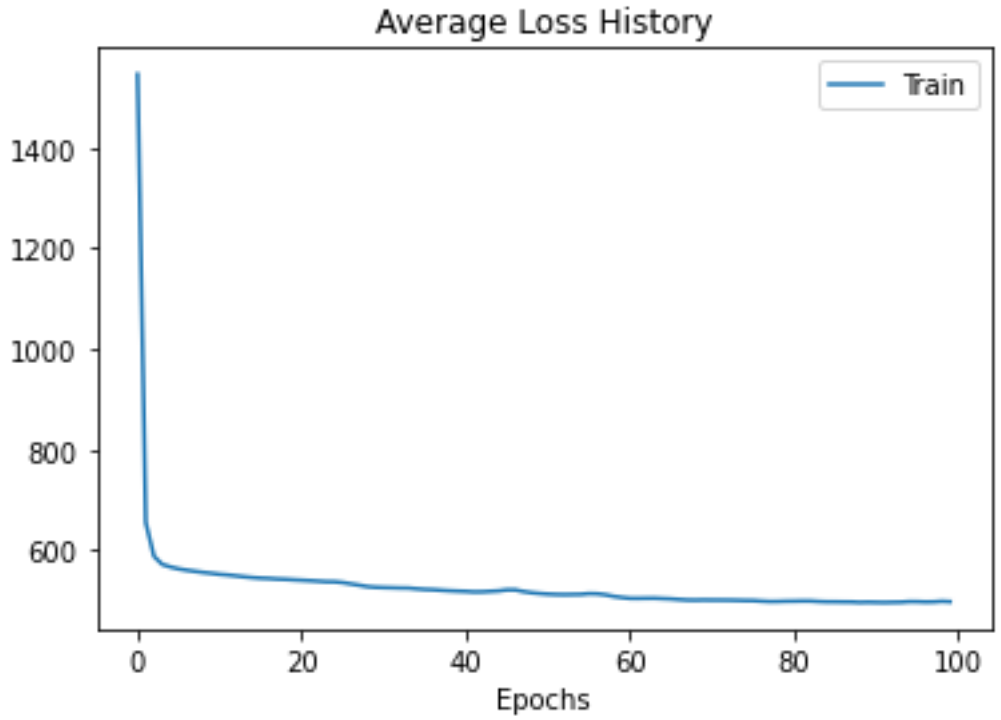
在 Lambda = 100 時

Amine_faces 的 reconstructed samples(左) 和 fake(右)



Amine_faces 的 interpolation

Amine_faces 的 loss curve



kl divergence 的目的是：讓 encode 出來的 latent code 的機率分布，盡可能接近 decoder decode 時要用的 latent code 的機率分布。如果你把 kl divergence 調太大，那網路會趨向讓 latent code 分布範圍增加，這樣怎麼 decode，跑出來的東西都會差不多。可以從實作中得出相同結論。

而所謂 posterior collapse，意思是解碼器太強大，它充分的學習到了每個輸入數據的特徵，使得 Encoder 編碼出來的隱分佈在離標準高斯很遠的時候就能被 Decoder 還原出來，這樣隱空間的存在就失效了，也就是上述所講的，跑出來的東西都會差不多。