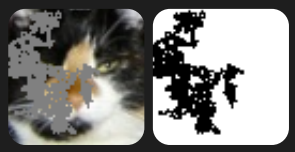
Lab5 MaskGIT for Image Inpainting - Experiment Score

313551055 柯柏旭

Part 1: Prove your code implementation is correct

1. Show iterative decoding

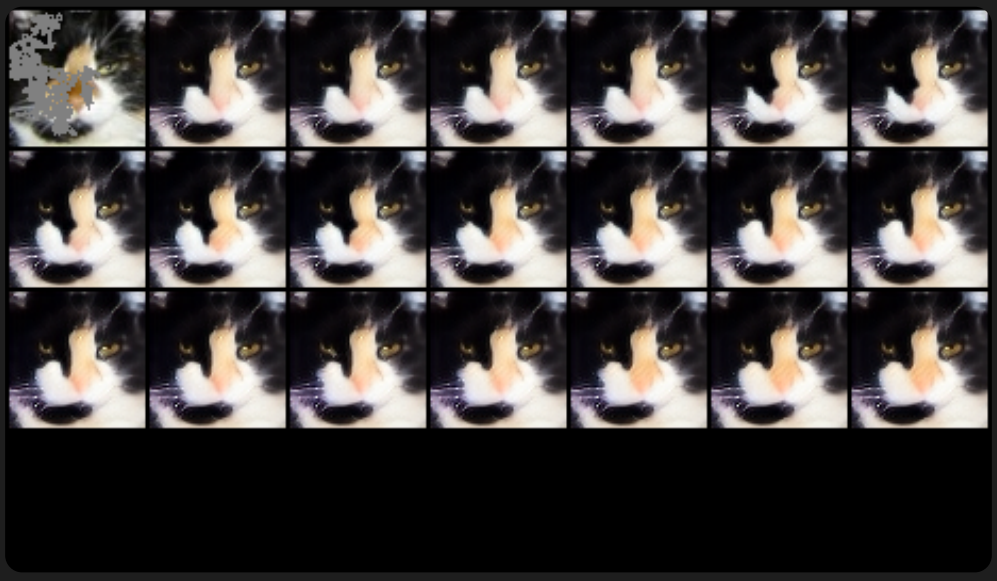
固定的設定 (image_000.png, total_iter = 25, sweet_spot = 20)



- cosine (FID: 44.2424566643580766)

(a) Mask in latent domain

(b) Predict image



- linear (FID: 44.29381567836265)

(a) Mask in latent domain

(b) Predict image

- square (FID: 43.10640596267979)

(a) Mask in latent domain

(b) Predict image

Part2: The Best FID Score

Screenshot

FID: 38.06985274794948

```
(DLP_lab) (base) hentci@gpu4:~/code/NYCU-DLP/lab5/faster-pytorch-fid$ python3 fid_score_gpu.py --predicted-path /home/hentci/code/NYCU-DLP/lab5/test_results --device cuda:0
747
100%|| 15/15 [00:00<00:00, 21.72it/s]
100%|| 15/15 [00:00<00:00, 30.88it/s]
FID:  38.06985274794948
```

Masked Images v.s MaskGIT Inpainting Results v.s Ground Truth

首先，透過以下程式將 `gt.csv` 轉換成圖片:

```python
import os
import pandas as pd
import numpy as np
from PIL import Image

# 讀取 CSV 檔案
csv_file = './faster-pytorch-fid/test_gt.csv'
data = pd.read_csv(csv_file, header=None)
num_images = data.shape[0]

# 創建資料夾來存儲圖片
output_dir = './ground_truth'
os.makedirs(output_dir, exist_ok=True)

# 將每一列轉換為圖片並儲存
for i in range(num_images):
    row = data.iloc[i].values

    # 將數據重塑為 64x64x3 的圖片格式
    img = row.reshape(64, 64, 3).astype(np.uint8)

    # 將 NumPy 陣列轉換為圖片
    img = Image.fromarray(img)
    if i == 0:
        continue
```

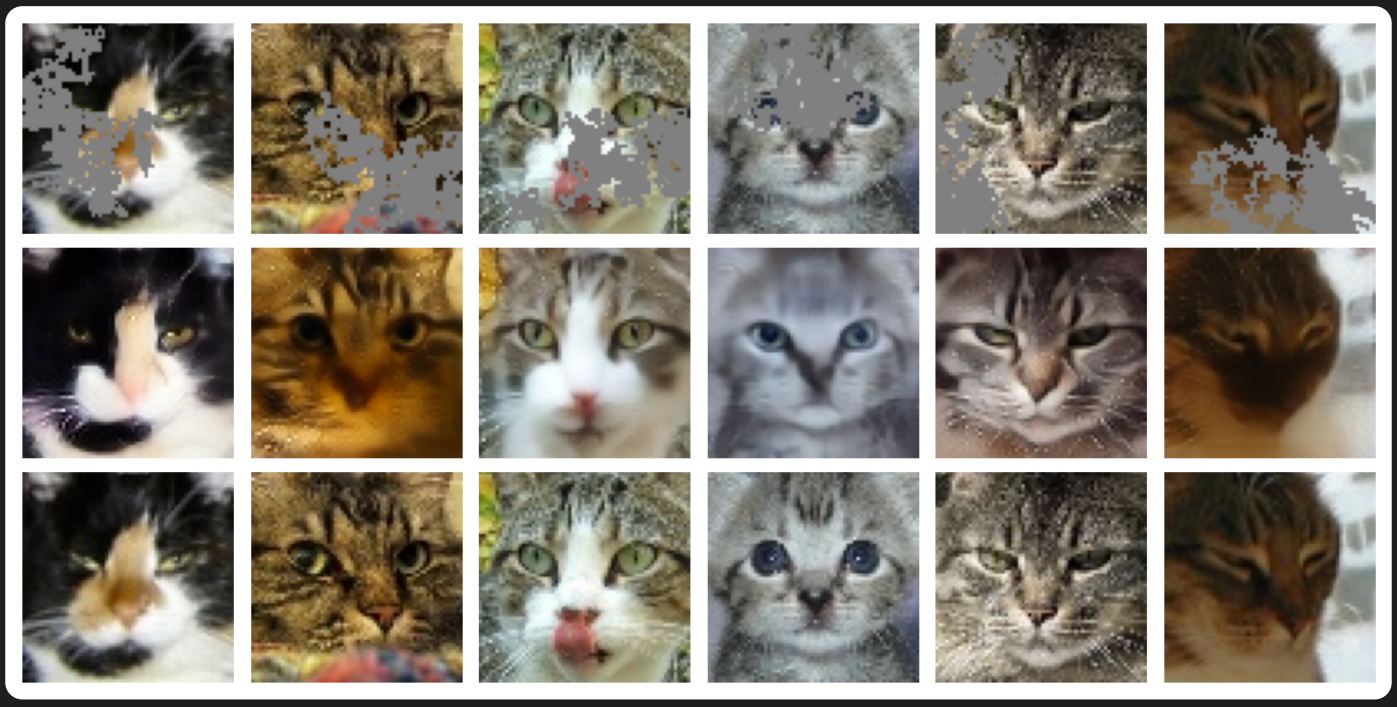
```
    # 儲存圖片
    img.save(os.path.join(output_dir, f'image_{i - 1:04d}.png'))

print(f"已經將 {num_images} 張圖片儲存到資料夾 '{output_dir}' 中。")
```

按照作業要求，生成以下比較圖片:

First row: Masked images

Second row: MaskGIT Inpainting Results

Third row: Ground Truth



The setting about training strategy, mask scheduling parameters, and so on

train 100 epochs，選取 valid loss 最小的 model

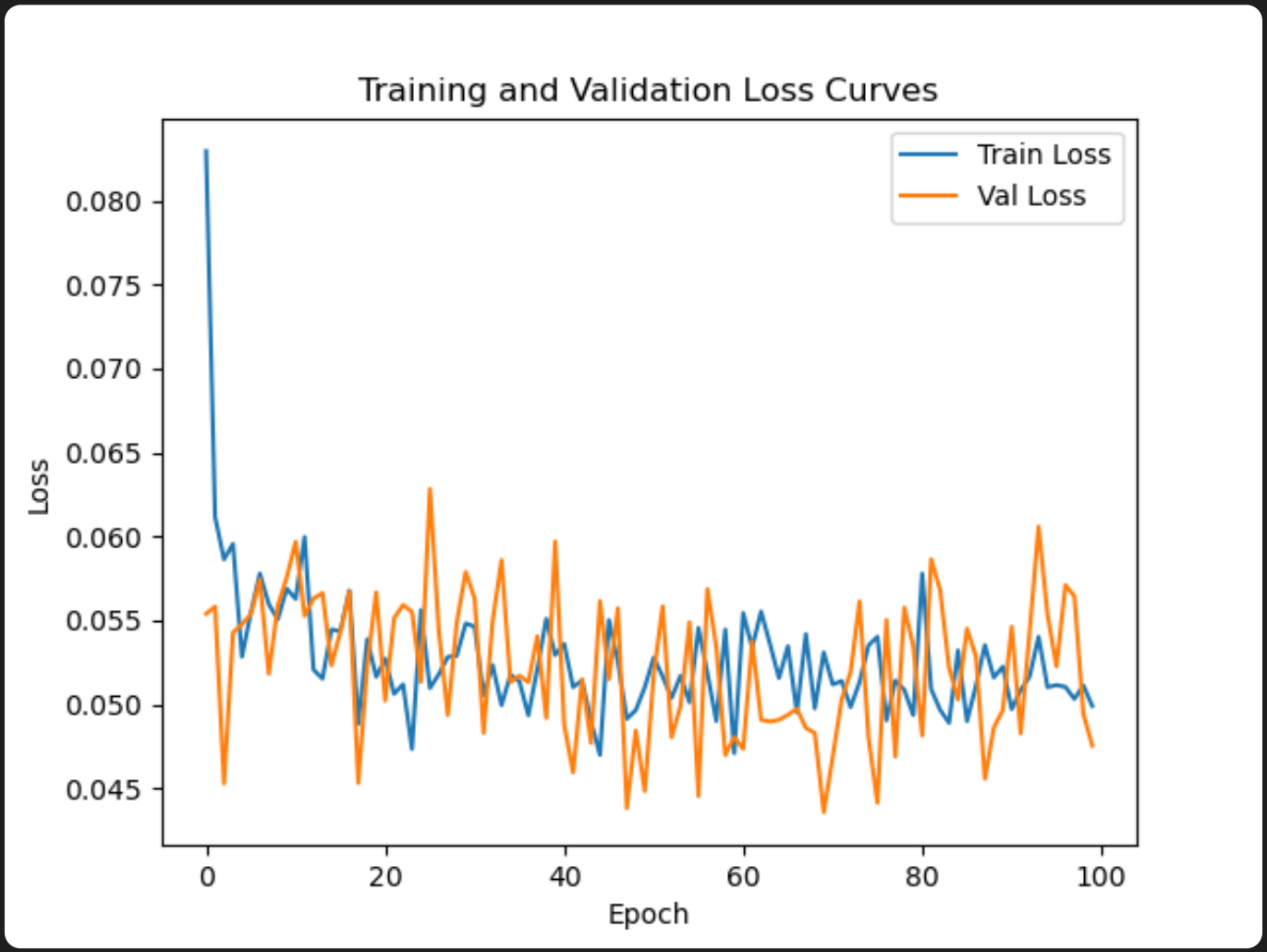
- Training hyperparameters

```
parser.add_argument('--epochs', type=int, default=100, help='Number
of epochs to train.')
parser.add_argument('--save-per-epoch', type=int, default=5,
help='Save CKPT per ** epochs(default: 1)')
parser.add_argument('--start-from-epoch', type=int, default=0,
help='Starting epoch number.')
parser.add_argument('--ckpt-interval', type=int, default=0,
help='Checkpoint interval.')
parser.add_argument('--learning-rate', type=float, default=1e-4,
help='Learning rate.')
```

- Inference hyperparameters

```
#MVTM parameter
parser.add_argument('--sweet-spot', type=int, default=2, help='sweet
spot: the best step in total iteration')
parser.add_argument('--total-iter', type=int, default=10, help='total
step for mask scheduling')
parser.add_argument('--mask-func', type=str, default='square',
help='mask scheduling function')
```

- loss curve


Training and Validation Loss Curves
Train Loss
Val Loss
Loss
0.080
0.075
0.070
0.065
0.060
0.055
0.050
0.045
0
20
40
60
80
100
Epoch