Stable Diffusion on OASIS Dataset

> The readme file should contain a title, a description of the algorithm and the problem that it solves(approximately a paragraph), how it works in a paragraph and a figure/visualisation.

Requirements (This should be removed when submitted)

1. The readme file should contain a title, a description of the algorithm and the problem that it solves(approximately a paragraph), how it works in a paragraph and a figure/visualisation.
2. It should also list any dependencies required, including versions and address reproduciblility of results,if applicable.3.
3. provide example inputs, outputs and plots of your algorithm
4. The read me file should be properly formatted using GitHub markdown
5. Describe any specific pre-processing you have used with references if any. Justify your training, validationand testing splits of the data.

Project Overview

Files

Non-Script

- `model_ckpt\*`: models' checkpoints.
- `log\*`: training log. (Can use `logger.py` to visualize it.)
- `*_vis\*`: visualization of images when training.
- `playground`: we write some code to figure out some algorithms.
- `README.md / report.pdf`: The file you current see.
- `tech_note.md / tech_node.pdf`: The main algorithm I used.

Common

- `util.py`: useful function, such as ssim, positional encoding.
- `module.py`: useful network module, such as resblock.
- `dataset.py`: get dataset from given folder.
- `logger.py`: script that can check the training log.

First Stage

- `model_VAE.py`: `VAE` and `VQVAE` model for first stage.
- `model_discriminator.py`: The GAN part when training `VAE` or `VQVAE`.
- `prestage_train.py`: Training Script for first stage.

Second Stage

- `pixelCNN.py`: Do pixelCNN when random generation from VQVAE model.
- `model_diffusion.py`: The model of stable diffusion. It's UNet.
- `stable_diffusion.py`: Do stable diffusion when random generation from `VQVAE` or `VAE` model.

Run

1. Requirements

```
einops==0.6.1
torch==1.11.0+cu113
torchvision==0.12.0+cu113
tqdm==4.66.1
Pillow==9.0.1
numpy==1.22.2
matplotlib==3.5.1
imageio==2.22.4
```

2. Change dataset, which is in `__init__` in `dataset.py`
3. For first stage
    1. run `prestage_train.py`. You can select the mode among `VAE` and `VQVAE` in `line 27` in this script.
4. For second stage
    1. TODO, collect indices data to another place.
    2. run `stable_diffusion.py`. If you want to use pixelCNN, you can run `pixelCNN.ipynb`

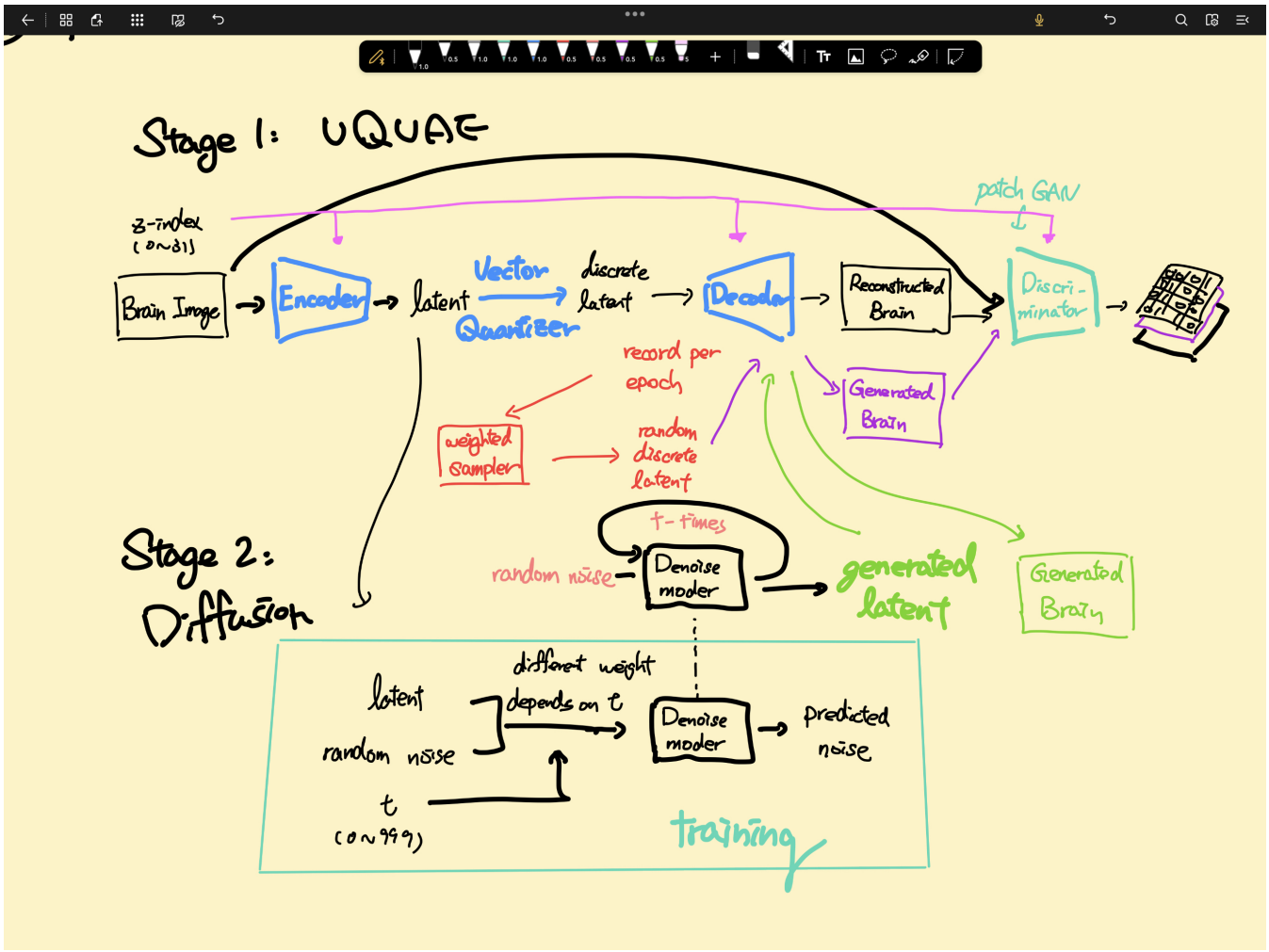
Results

Diffusion Process GIF

Algorithm Overview

1. I've applied VAE/VQVAE to the OASIS dataset, achieving a high SSIM score of approximately 0.78.
   Here's your description with improved grammar:
2. In the first stage of training, which involves training an autoencoder on the OASIS dataset, I have explored several enhancements:
    1. Introduced a GAN component to improve image reconstruction clarity (VAEGAN).
    2. Utilized a transformer to predict the latent space (VQGAN).
    3. Integrated an auxiliary loss to train the decoder (or generator) using random latent vectors as input due to the limited dataset availability.
    4. Employed the z-index of brain data as a conditioning factor to teach the model how to generate images with specific conditions (Pix2pix or conditional GAN).
    5. Implemented a cyclical annealing schedule for VAE to prevent mode collapse.
    6. For VQVAE, I employed a weighted sampler to sample the discrete latent space and used it to train the decoder and discriminator within the sampled space.
3. After training the VQVAE, I applied DDPM (Denoising Diffusion Probabilistic Models) to the latent space, which is the core idea behind stable diffusion.
    - Note that stable diffusion comprises with two key contributions: DDPM on the latent space and cross-attention across different modalities. However, since the OASIS dataset lacks of other conditions, we did not implement cross-attention in this repository.

Total Flow Chart



The main concepts and related workds are in `tech_note.md`