Report of Manga character generation with diffusion models

Francesca Visini (502176)- Alice Menna (502172)

Introduction:

In this project we are going to implement a diffusion probabilistic model for generating manga characters' faces.
Diffusion models are generative models. The idea is to collect a training set of samples with a given distribution and generate data that resemble the same distribution.
Diffusion models are based on stochastic processes, in particular they are based on a forward and a reverse process. The forward mechanism deals with cleaning data and with a number T of steps. At each step we add some gaussian noise until we completely lose the initial information of the image. At this point, the reverse process occurs. Starting from a situation of complete noise, we remove it step by step to obtain the clean data.
The process is stochastic because the initial point of the reverse process is different anytime and, as a consequence, we will obtain different results.

Set up and Data:

We use a dataset containing 12000 images of manga character's faces, each one already scaled down to 64×64 pixels.
The necessary libraries are imported at the beginning, including torch for tensor operations, torchvision for handling datasets and image transformations, matplotlib for visualization.
Then we add some transformations: reduce the side of image to speed up training, random horizontal flip for data augmentation, transformation to tensors and normalization with 0,5.
The data loader is constructed with a batch size of 32 and applying shuffle and using 2 workers in parallel.

Model:

First of all we define the forward process where we add gaussian noise to a starting clean image taken from the training set. We perform the process using 300 time steps.
Adding the same amount of noise at each time step is not optimal, so it will be added more noise toward the end of the process, considering the following formula:

$$x_t = \sqrt{\alpha_t} x_{t-1} + \sqrt{1 - \alpha_t} \epsilon$$

After the forward process, we compute the reverse process. In order to do it, we use a neural network. The architecture is a Unet-like CNN with residual and skip connections, and with an attention mechanism. It is provided in the unet.py module.
We train the algorithm using 20 epochs. In each epoch, it processes batches of images, applying random noise at random time steps, and trains the neural network to predict this noise.

After training we generate new images using sampling. We start from random noise and gradually refining it into reliable images.

Questions:

Question 1:

The number of time steps in a diffusion process plays a critical role in both the training and sampling phases. Changing the number of time steps can significantly impact the performance, quality of the generated images, and computational requirements.
Increasing the number of time steps can cause:

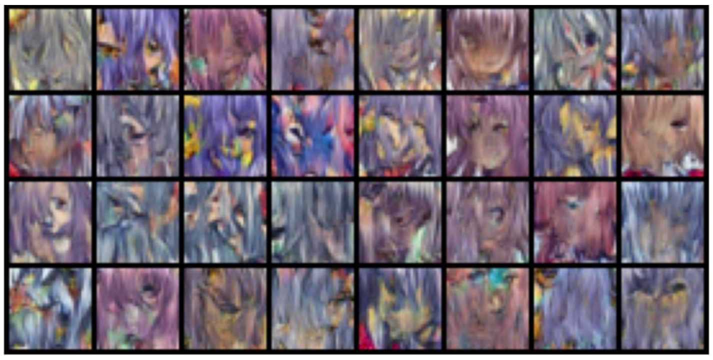
- Training time incrementation due to more iteration of each image
- Higher memory usage
- Higher quality of generated images

Decreasing the number of time steps can cause:

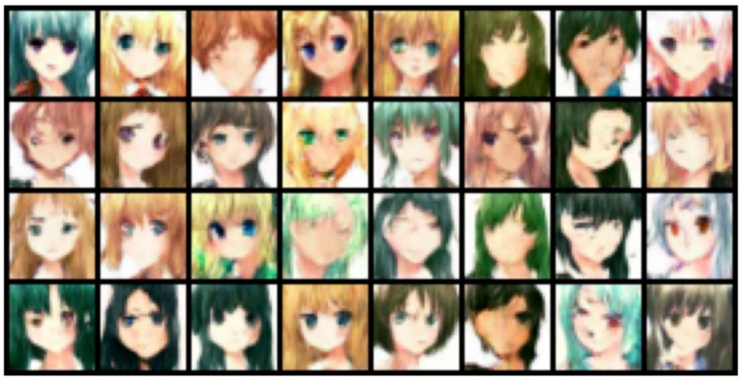
- Reduction in training time, making the process faster.
- Lower quality of generated images
- Higher chance of introducing inaccuracies because the denoising process is faster

In general, increasing the number of time steps enhances the quality of the images making them more realistic, but it takes longer training and sampling time. On the other hand, decreasing the number of time steps, leads to a lower memory usage and a faster training and sampling time but produces low quality images and reduces the accuracy.
We try with 100 and 500 time steps.

- 100 time steps



- 500 time steps



Question 2:

The initial noise is typically drawn from a standard normal distribution. The choice of initial noise directly influences the variety and quality of generated samples. If the initial noise is not included and, for example, it is set constant, the generated samples don’t show diversity because the process would start always from the same initial state, making the output similar or identical. To disable the initial noise, we set it to zero.
The step noise is estimated using a neural network trained to predict the noise added during the forward diffusion process. If the step noise is disabled and removed in each backward step, we will have a poor reconstruction of the original image, as the model would not correctly reverse the noise added during the forward process.