Face Generation with Denoising Diffusion

1st Bernardo Taveira
dept. Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden
taveira@chalmers.se

2nd Yiyun Xia
dept. Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden
yiyun@chalmers.se

***Abstract*—In this project, ee trained a denoising diffusion model to generate face images from pure noise. We scaled the training to make it feasible on a personal computer, and managed to generate interesting results of reasonable fidelity for 32×32 and 64×64 image outputs based on the CelebA [5] dataset. We also investigated on the architecture and the training process of the model, and presented some validations of its mechanisms.**

***Index Terms*—diffusion models, image synthesis, face generateion, denoising, UNet**

I. Introduction

In the field of image synthesis, there have been various deep generative models such as GANs [2], VAEs [4], autoregressive models [6], and diffusion models [10]. Among them, diffusion-based models have arguably outperformed other models in many image synthesis tasks [1] [8], and benefited from its training stability as well as the likelihood-based mechanism for better scalability and variability. Recently they have grown a lot in popularity with the introduction of implementations such as Dall-E 2, Imagen and Craiyon.

The aim of this project is to explore the denoising diffusion model by implementing it in a reasonably scaled problem, which requires training the denoiser and generate new images that appear to be from the same distribution as the training dataset. The images of faces (CelebA [5]) was chosen as the dataset, since faces have intricate shapes and us as humans can very easily point out when an image of a face is not real.

II. Related work

Image generation has been a challenge in deep learning for over a decade. Generative adversarial networks had been the most prominent contender for the task until recent years due to the ability to produce high resolution and believable images. However they prove to be extremely difficult to train. Vanishing gradients is very common in GANs as well as mode collapsing which is when the generator finds a very plausible output and learns to always go for that same or similar output.

Diffusion models have been a prominent topic in the deep learning scientific community the past years. Most notably after a paper from OpenAI [1] was published proving the capability of these models in surpassing GANs in image generation.

However, denoising diffusion models have been experimented on for a much longer period, dating back to 2015 [10]. Since then a lot of improvements have been made on them. One big improvements was made in [3] where the UNet used for the denoising step was perfected by adding a lot of new concepts such as attention and residual connections.

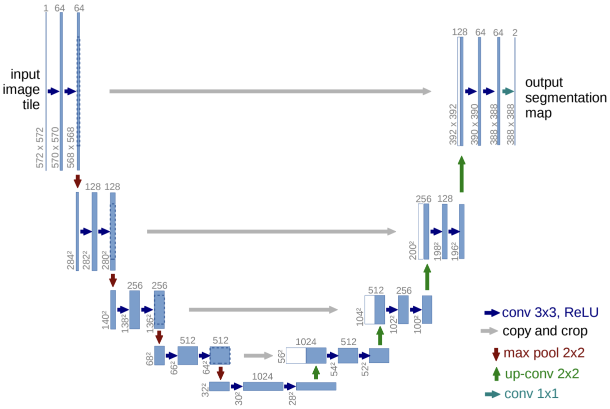

Fig. 1. Representation of UNet architecture

III. Method

A denoising UNet is used for the noise prediction. This model was first presented in [9] and then perfected in [3]. It works similarly to an autoencoder, first downsampling the input and after some layers in latent space it upsamples again to produce an output. This structure is illustrated in figure 1.

The downsample consists of multiple small downsample stages. Each stage starts off with two ResNet layer or ConvNeXT layers [7] followed by a group normalization, an attention block, a residual connection and finally a downsample opperation consisting of a simple convolution layer.

Both ResNet and ConvNeXT [7] were experimented on and with the little training done no significant differences in performance were found.

Position embedding is used to encode the noise level into the the denoising step. As described in the DDPM paper [3] sinusoidal position embedding inspired on the Transformer architechture [11] is used for this network.

In the middle of the UNet, now in latent space, two more ResNet or ConvNeXT layers are applied with an attention and a residual block.

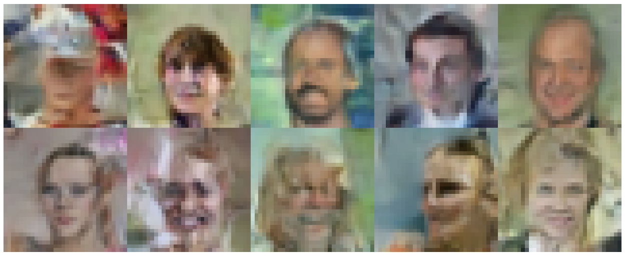

Fig. 2. Generated images from pure noise after 5 epochs of training and 200 denoising steps

The umpsample works very similarly to the downsample with the only difference being the use of a transposed convolution to upscale the tensor.

IV. Results and discussion

As mentioned before the CelebA [5] dataset was used for training. Since these models can require a massive number of training steps to achive reliable and plausible results, first smaller training loops were executed to find optimal parameters. Our conclusions on these will be presented here, however one must keep in mind that a reliable metric to evaluate the results was not created so quantifying the improvements becomes quite subjective to looking at examples of randomly generated faces.

Taking into account the limited resources available and the length of training needed to further investigate these fascinating models, Google's free Colab platform was used for small tests and comparisons between parameters. It was not used for long training sessions due to the limit in runtime it has. A personal computer with a NVIDIA gtx1060 6Gb was then used to continuously train the multiple models tested on the Colab platform for multiple days at a time. Additionally, to test the possibility of expanding this model to 64x64px images, the Google Cloud T4 GPU had to be used given that the model itself would take over 13Gb of VRAM on the GPU, requiring then a more suitable GPU than our local one.

These 3 training setups allowed us to always train at least two models at the same time and for multiple days in a row while having also a platform to do small tests and prepare new setups, therefore optimizing the usage of the two good GPUs available.

A. Denoising Steps

One of the first big questions that arise is how many denoising steps are needed to achieve optimal results.

In the Jonathan Ho paper [3] for a similar objective but higher dimension image of 256x256px 500 steps were used. On the other hand, in the paper [1] where DM are proved more effective than GANs at generating realistic images, 250 steps were used to generate images of size 128x128px.

Given this first analysis of previous papers, 200 steps were fist chosen and proved successful as it can be seen in figure 2 after only 5 epochs.

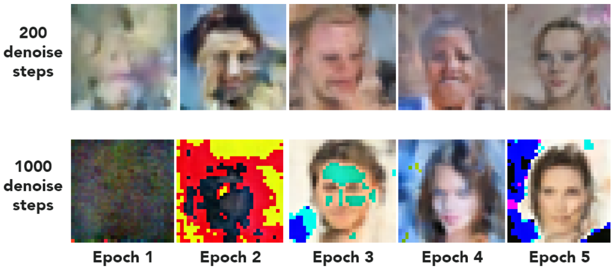

Fig. 3. Comparison between 200 and 1000 denoise steps for first epochs of training

A further investigation of this parameter was executed. First reducing to 100 steps which produced sub optimal results. Then increasing this number to 1000 steps to try to understand the effect of the number of steps on the results.

The most noticeable change was how the increase of the number of steps required a much higher number of epochs to get acceptable results (figure 3). Probably due to the fact that the amount of positional embedding it needs to learn has also increased. Despite this extra training required the model seemed to converge to a face always. The same can not be said for the 200 denoising steps model. Some color artifacts appeared in the images for the 1000 denoising steps model and only disappeared after even more training.

With this in mind it was established that 200 steps is a good efficient value. By training it further it is possible to obtain realistic faces.

B. Process Variance Schedule

The authors of the original Denoising Diffusion Model [3] set the variance as a linear interpolation between two values following the step number. However, it was later shown [1] that a cosine distribution gave better results.

In our case to try to gain some understanding on this, both were tried for about 10 epochs without any clear difference in the outputs. Additionally a quadratic and sigmoid schedule were also experimented on but without much success and with worse results. One can assume that the differences between the cosine and linear schedules could become evident when training for much longer where one could result in near perfect results.

Given our limited computational resources and time it was decided to follow the original paper [3] and use a linear schedule.

C. Loss Function

For the loss function we started of using the same used in the paper. Known as tmooth L1 loss or Huber it simply consists of an L1 losso with a quadratic element that smooths near the value o 0 error. Since this loss produced good training progress it was used throughout the entire project.

D. Learning Rate

For the learning rate we found that 10^{-5} produced good enough results and reasonably fast convergence. We noted later

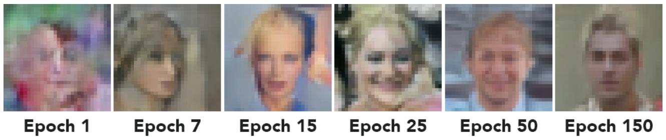

Fig. 4. Generated images from various stages of training

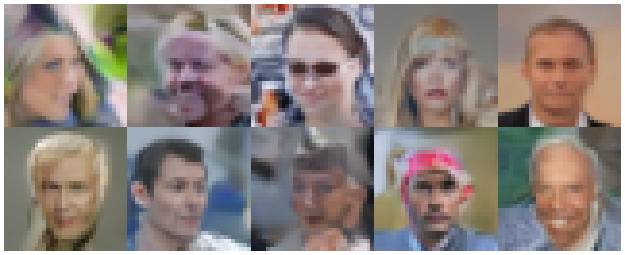

Fig. 5. 10 samples generated from pure noise of the fully trained model

for a long training that reducing it to 10^{-6} after about 100 epochs could help improve the results.

E. Training

After having established the parameters described before the model was trained for 150 epochs, taking over 30 hours. It was trained 25 epochs at a time, saving the model at each batch of epochs. In the last 50 epochs further investigation of the learning rate was conducted where it was trained with different learning rates from the same starting point and compared to evaluate if the reduction of learning rate would help the model converge faster to more plausible images.

The results in multiple stages of the training can be seen in figure 4.

The trained model can be evaluated by generating a batch of images from pure random noise. A sample of these can be visualized in figure 5 The denoising process can be explored by saving the images in throuthout the 200 steps. The figure 6 shows this denoising process clearly.

F. Increase in image size

An attempt to train for higher size images was made. For 64x64px images a GPU with at least 15Gb was required. The learning rate had to be increased to 10^{-3} and as expected the training required substantially more time to train. After initial setups and tests we were only able to train for 66 epochs which took over 46 hours of continuous training.

The results can be analysed in figure 7. It can be seen that there is a potential for very high resolution images that could fool a human into believing they are real. However the results are not consistent and some samples are sub-optimal. More training steps could most likely fix these issues. If not enough a small increase in the number of denoising steps could make the model converge more consistently to a believable face but again would also require more training time.

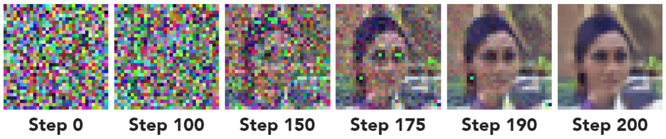

Fig. 6. Diffusion process from pure noise image to full 200 denoising steps

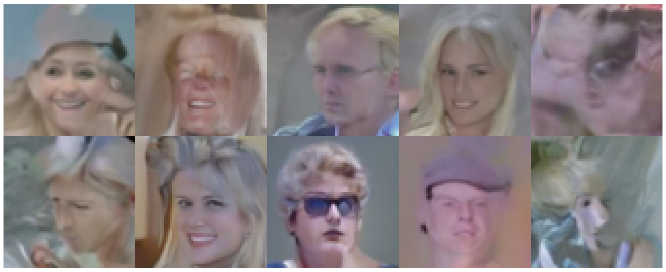

Fig. 7. 10 samples generated from pure noise of the trained 64x64px model

V. Conclusions

We have managed to train a denoising diffusion model to generate faces from noise with success. Not only have we been able to get interesting results but also gain further understanding of the architecture and the training process of this diffusion process.

One possible big improvement would be to use an auto encoder to encode a higher resolution image into latent space and then use this UNet model to execute the multiple steps of denoising in the latent space just as described in the Latent Diffusion paper [8]. This could enable us to still train with limited resources and increase the image dimension although would require either use a pre-trained autoencoder or train one ourselves.

References

[1] P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. *CoRR*, abs/2105.05233, 2021.

[2] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks, 2014.

[3] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *CoRR*, abs/2006.11239, 2020.

[4] D. P. Kingma and M. Welling. Auto-encoding variational bayes, 2013.

[5] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[6] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[7] Z. Liu, H. Mao, C. Wu, C. Feichtenhofer, T. Darrell, and S. Xie. A convnet for the 2020s. *CoRR*, abs/2201.03545, 2022.

[8] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models, 2021.

[9] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.

[10] J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *CoRR*, abs/1503.03585, 2015.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.