Inverse problems with Pretrained Diffusion models

Venkat S. Rao

Abstract—Study use cases of diffusion models. Use 5 test cases to quantitatively and qualitatively understand where diffusion models can be used.

Index Terms—Computational Photography, Diffusion, Deep learning, inpainting, denoising, deconvolution

1 Introduction

Solving Inverse problems such as deconvolution and inpainting have many applications in computational photography. Deconvolution can be used recover blurry images caused by camera or subject movement. Inpainting can be used to remove unwanted objects or soviet leaders who have lost favor with the state.

Diffusion models can be used to denoise an image. We can use their noise prediction outputs as priors to solve inverse problems. As these diffusion models are the state of the art, we want to gain a better understanding of their applications. It will be great to understand where they are useful.

2 Related Work

For inpainting and deconvolution, we have many methods unrelated to Diffusion and also many variations of diffusion that can be used.

2.1 Traditional Computational Photography Methods

Inpainting can be performed via traditional computational photography using algorithms such as the Fast Marching method [1] and another method based on fluid dynamics and partial differential equations [2].

2.2 GANs and Variational Autoencoders

GANs and Variational Autoencoders enjoyed the spotlight prior to diffusion models for generating synthetic images. Diffusion models perform better than GANs since they more stable during training. And they are better than Variational autoencoders since they produce more real and less blurry images.

2.3 Diffusion

Diffusion models generate images iteratively from noise. They transform an image into noise through a forward diffusion process and then learn to reverse this process to generate new, high-quality images from noise.

- *M. Shell is with the Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, 30332.*
  E-mail: see http://www.michaelshell.org/contact.html
- *J. Doe is with Anonymous University.*

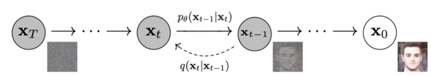

Fig. 1. DDPM process

2.4 DDPM

Denoising Diffusion Probabilistic Models(DDPM) [3] gradually add noise to an image during training (a forward diffusion process) and then learning to reverse this process (a denoising process) to generate new images from pure noise. This markov chain, see 1 uses a controlled noise schedule. This method provided state of the art results when released in 2020.

DDPMs use a Deep learning models with a Unet architecture [4] to predict noise. But any model can be used to provide a noise prediction.

2.4.1 Langevin dynamics

Langevin dynamics based sampling adds a noise term when sampling a prior and this ensures unique solutions for each inference.

2.5 SDEdit

Stochastic Differential Editing [5] provides a way to balance generating a real looking image while the maintaining fidelity with the original ground truth image. The amount of noise applied, as measure in timesteps, is a hyperparameter that can be tuned.

2.6 ScoreALD

ScoreALD [6] was initially created for MRI images to decrease radiation exposure for patients. It could interpolate images based on input data. It adds a condition based on the gradient of the difference between a prior and the measurement. It scales this prior by annealed Langevin Dynamics factor.

2.6.1 Annealed Langevin dynamics

Langevin dynamics models often have trouble converging. ScoreALD introduces an annealing factor to the scaling term. The annealing factor decreases as the timesteps increase. In the paper, the authors were able to get better results using this method.

2.7 Diffusion Posterior Sampling(DPS)

Diffusion Posterior Sampling(DPS) [7] modifies the standard diffusion sampling process by conditioning on the likelihood of the observed data, ensuring that samples remain consistent with given measurements. This is done by adjusting the reverse diffusion process to account for the posterior distribution. Additionally, it uses a predicted non noisy initial image to condition rather than the current posterior.

3 METHODS

We used five different methods to understand various methods of applying diffusion models to inverse problems.

3.1 Single-Pass Image Denoising

We look at multiple images for three inverse tasks: deconvolution, inpainting with random masks applied to the whole image and inpainting a random mask.

Since we can choose how large a timestep we can take, we compare all three methods using three different timesteps.

3.2 Unconditional Generation with DDPM

This is the first method, where we use a iterative markov chain for denoising. A basic DDPM implementation is used to generate images.

We explore unconditional image generation with DDPM. However, since the model was trained on a dataset of faces, it will only generate faces.

We judge the image qualitatively, for how close the generated person looks real. We look for artifacts that indicate the image is fake.

3.3 SDEdit

Since SDEdit allows us to to choose the amount of noise as a hyperparameter, we look at image quality at multiple timesteps and judge how realistic a generated image looks. We also compare time to generations. Additionally, we want to ensure all added artifacts, such as a mask, have been removed.

3.4 Inverse Problems - ScoreALD vs DPS

We compare ScoreALD and DPS to judge how they perform. We compare them on the same three tasks as SDEdit: deconvolution, inpainting with box masks and random masks.

For DPS, we can also tune the scale parameter to find the best scaling factor.

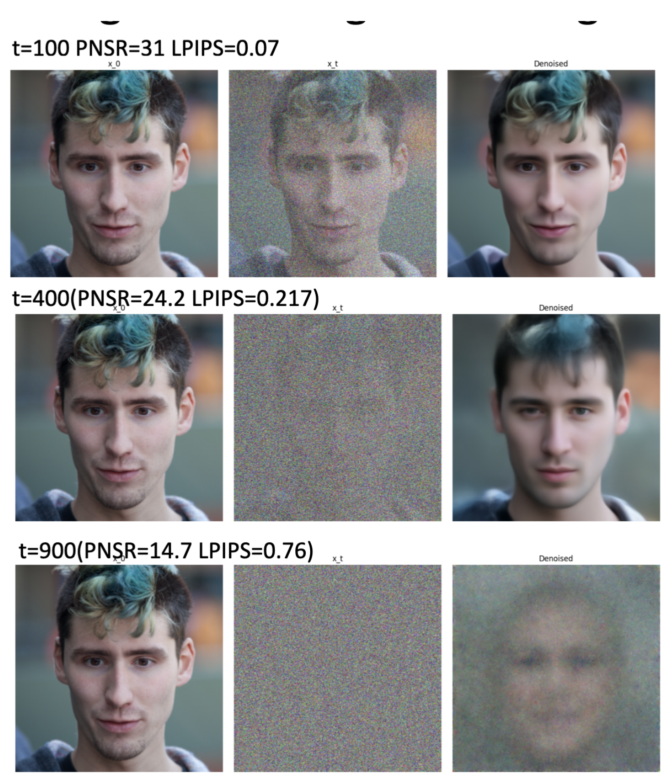

Fig. 2. Single-Pass Image denoising

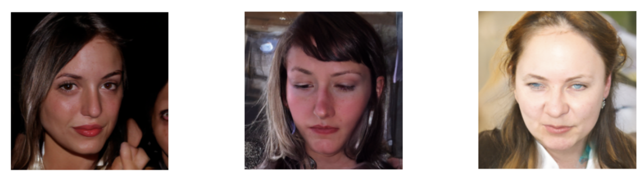

Fig. 3. Unconditional Generation

4 EXPERIMENTAL RESULTS

4.1 Single-Pass Image Denoising

See figure 2.

The less noise that is added, the easier it is to recover the original image. As we go past 500 timesteps, the generated image loses any fidelity to the ground truth.

Quantitatively, we see similar results. If we went past 500 timestamps, the image would no longer have any resemblance to the ground truth and provide results with PSNRs less than 20.

At t=400, we can see the image has quite a lot of noise, but can still recover the original.

4.2 Unconditional Generation with DDPM

In figure 3, we see three images that were unconditionally generated. All faces could be real people.

4.3 SDEdit

4.3.1 Deconvolution

In figure 4 we can recover a deconvolved image for a timestep between 200 and 500.

Even at timestep 200, this process cannot reproduce a great image. At 900, a completely new person is generated.

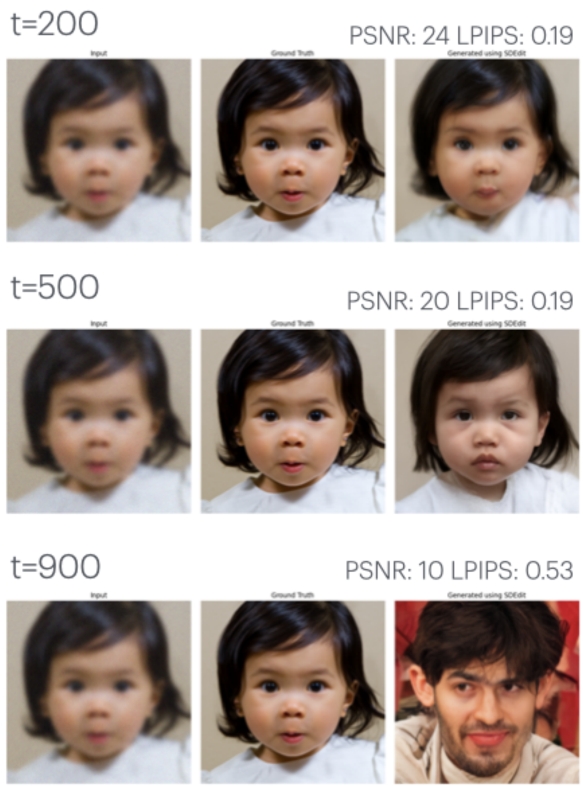

Fig. 4. Deconvolution with SDEdit

4.3.2 Inpainting - Mask

In figure 5 we see SDEdit is not a great method to recover masked areas at any timesteps. Even at low and mid timesteps, the mask is not removed. At timestep 500, the mask is incorporated into the face and the generated image look like cyborg child.

4.3.3 Inpainting - Random

In figure 6 we recover cartoonish images at timesteps less than 500. At timestep 900, the recovered image has no relation to the Ground truth.

Even at timestep 200, the process cannot reproduce a great image. Given the entire image was masked we maybe should not expect great results.

4.4 Inverse Problems - ScoreALD vs DPS

4.4.1 Deconvolution

In Figure 7 ScoreALD produces a very saturated image. DPS produces a high quality reproduction.

The PSNR score of each is pretty close but the LPIPS, where lower is better, is almost half for DPS.

4.4.2 Inpainting - Mask

In Figure 8 ScoreALD has produced an oversaturated image with a weird looking eye. DPS, on the other hand, seems to be have done well on this task.

Qualitatively, DPS image is very close to ground truth, but we can see a different right ear ring compared to the ground truth.

PSNR is 22 for ScoreALD and 35 for DPS. This is the highest PNSR score for any of the methods.

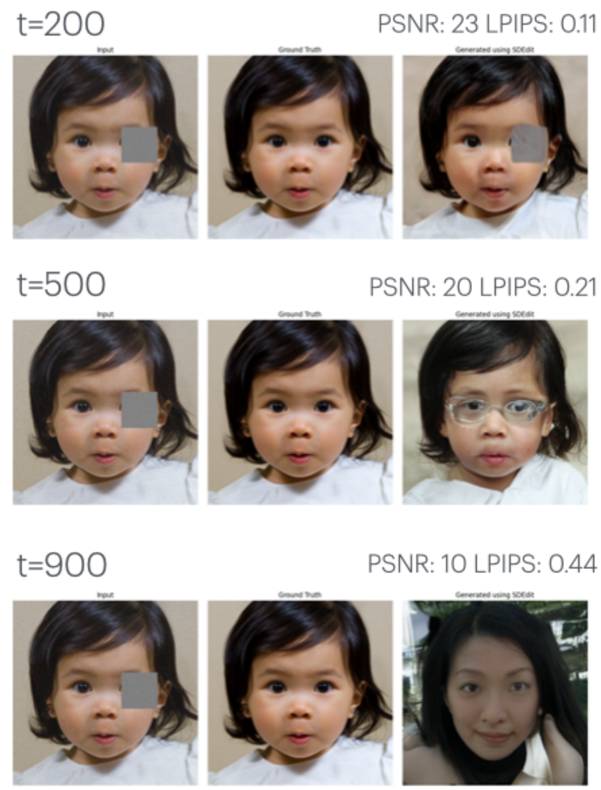

Fig. 5. Inpainting- Box mask with SDEdit

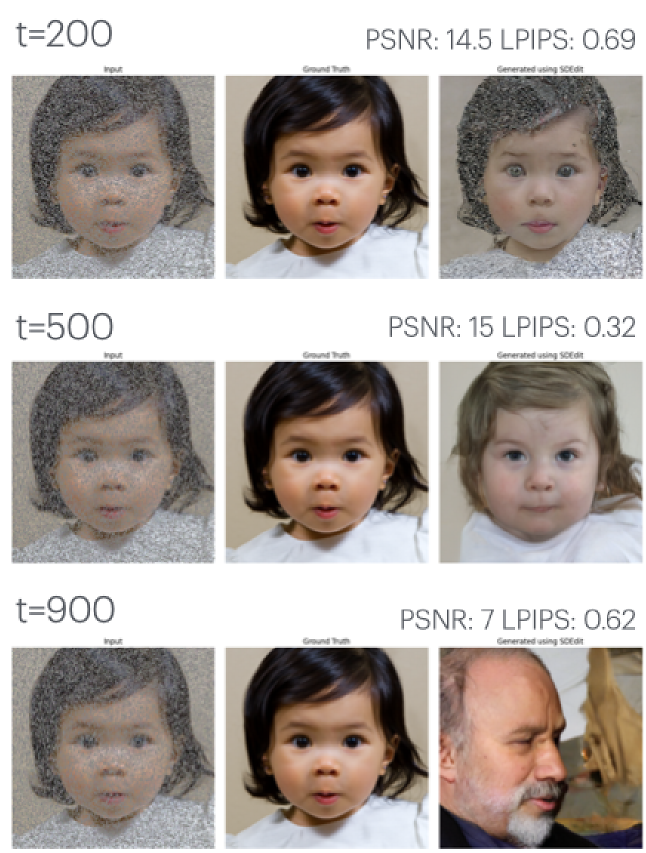

Fig. 6. Inpainting - Random mask with SDEdit

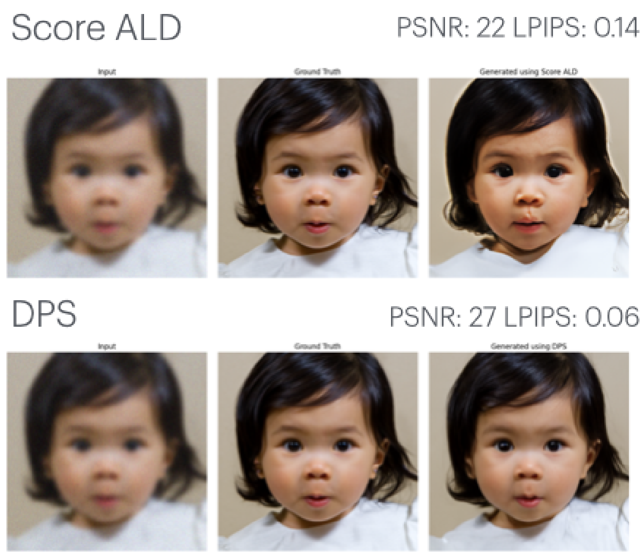

Fig. 7. Deconvolution

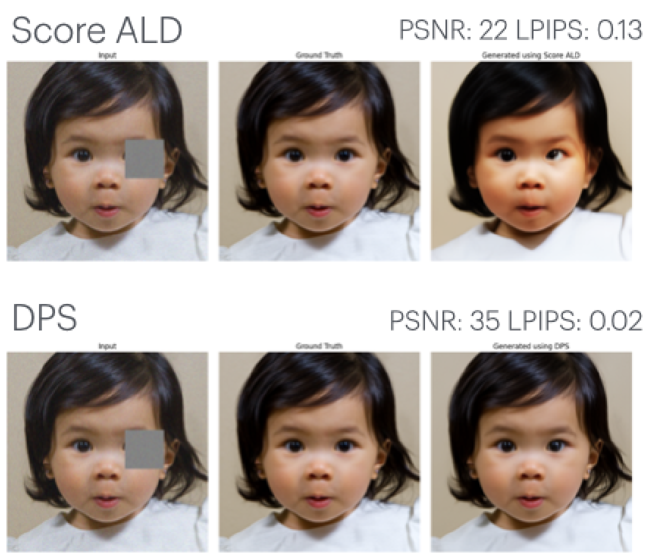

Fig. 8. Mask Inpainting

4.4.3 Inpainting - Random

In Figure 9 we see both methods can somewhat recreate the ground truth. Both methods seem to have recreated a different child. DPS performs much worse than ScoreALD for this task.

PSNR is 20 for ScoreALD but qualitatively the algorithm has performed much worse.

5 Discussion

For the inpainting task with a box mask, all three methods, SDEdit, ScoreALD and DPS, change the unmasked portion of the image changes. This is probably unacceptable in a commercial product like Lightroom.

In addition to the qualitative and quantitative results, running these models takes quite a long time. It also requires GPUs to run.

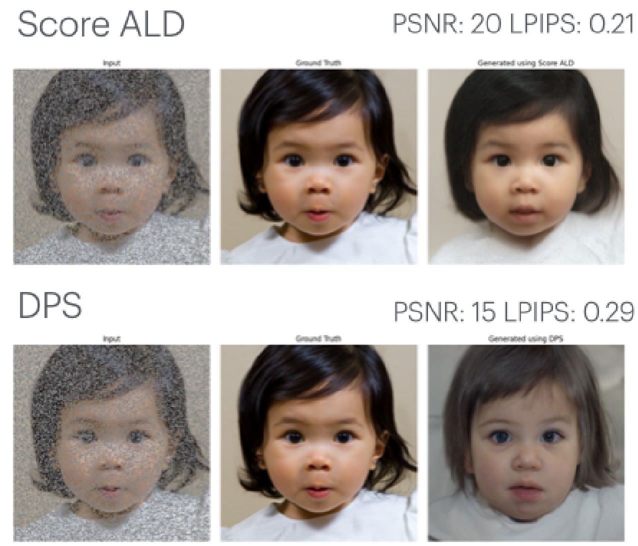

Fig. 9. Random Inpainting

6 Future work

Some other diffusion methodologies like stable diffusion [8] and diffusion model that replace Unet with a Vision transformer [9] should also be looked at for these tasks.

Another idea might be to combine a traditional computational photography method with diffusion.

7 Conclusion

Qualitatively, it seems all methods cannot impaint or deconvolve images without issues.

Quantitatively, DPS has the best scores and future work should focus on improving it.

The methods explored in this paper do not perform well enough given heavy resource usage. We also need to study different and follow on methods.

Acknowledgments

The author would like to thank the EE367 staff and especially Axel Levy for providing support while working on this project.

References

[1] A. Telea, "An image inpainting technique based on the fast marching method," *Journal of graphics tools*, vol. 9, no. 1, pp. 23–34, 2004.
[2] M. Ebrahimi and E. Lunasin, "The navier–stokes–voight model for image inpainting," *The IMA Journal of Applied Mathematics*, vol. 78, no. 5, pp. 869–894, 2013.
[3] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.
[4] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*. Springer, 2015, pp. 234–241.
[5] C. Meng, Y. He, Y. Song, J. Song, J. Wu, J.-Y. Zhu, and S. Ermon, "Sdedit: Guided image synthesis and editing with stochastic differential equations," *arXiv preprint arXiv:2108.01073*, 2021.

[6] A. Jalal, M. Arvinte, G. Daras, E. Price, A. G. Dimakis, and J. Tamir, "Robust compressed sensing mri with deep generative priors," *Advances in Neural Information Processing Systems*, vol. 34, pp. 14 938–14 954, 2021.

[7] H. Chung, J. Kim, M. T. Mccann, M. L. Klasky, and J. C. Ye, "Diffusion posterior sampling for general noisy inverse problems," *arXiv preprint arXiv:2209.14687*, 2022.

[8] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 684–10 695.

[9] W. Peebles and S. Xie, "Scalable diffusion models with transformers," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 4195–4205.

APPENDIX

.1 Task 1.1

$$x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}z$$

$$\alpha_t = 1 - \beta_t$$

$$\bar{\alpha}_t = \Pi a_t$$

$$x_{t-1} = \sqrt{1-\beta_{t-1}}x_{t-2} + \sqrt{\beta_{t-1}}z$$

$$x_t = \sqrt{1-\beta_t}(\sqrt{1-\beta_{t-1}}x_{t-2} + \sqrt{\beta_{t-1}}z) + \sqrt{\beta_t}z$$

$$x_t = \sqrt{1-\beta_t}\sqrt{1-\beta_{t-1}}x_{t-2} + \sqrt{1-\beta_t}\sqrt{\beta_{t-1}}z + \sqrt{\beta_t}z$$

$$x_t = \sqrt{\alpha_t}\sqrt{\alpha_{t-1}}x_{t-2} + \sqrt{1-\beta_t}\sqrt{\beta_{t-1}}z + \sqrt{\beta_t}z$$

If we recursively replace the x_{t-2} term, we would get:

$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}z$$

.2 Task 1.2

$$\hat{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t + (1-\bar{\alpha}_t)s_\theta)$$

$$x_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\alpha_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\hat{x}_0$$

$$x_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\alpha_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}(\frac{1}{\sqrt{\bar{\alpha}_t}}(x_t + (1-\bar{\alpha}_t)s_\theta))$$

$$x_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\alpha_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t\sqrt{\bar{\alpha}_t}}x_t + \frac{\sqrt{\alpha_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t\sqrt{\bar{\alpha}_t}}(1-\bar{\alpha}_t)s_\theta$$

$$x_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\alpha_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t\sqrt{\bar{\alpha}_t}}x_t + \frac{\sqrt{\alpha_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t\sqrt{\bar{\alpha}_t}}(1-\bar{\alpha}_t)s_\theta$$

$$x_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{(1-\alpha_t)}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}x_t + \frac{(1-\alpha_t)}{\sqrt{\alpha_t}}s_\theta$$

$$x_{t-1} = (\alpha_t(1-\bar{\alpha}_{t-1}) + 1 - \alpha_t)\frac{x_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}} + \frac{(1-\alpha_t)}{\sqrt{\alpha_t}}s_\theta$$

$$x_{t-1} = (\alpha_t - \alpha_t\bar{\alpha}_{t-1} + 1 - \alpha_t)\frac{x_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}} + \frac{(1-\alpha_t)}{\sqrt{\alpha_t}}s_\theta$$

$$x_{t-1} = (1-\bar{\alpha})\frac{x_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}} + \frac{(1-\alpha_t)}{\sqrt{\alpha_t}}s_\theta$$

$$x_{t-1} = \frac{x_t}{\sqrt{\alpha_t}} + \frac{(1-\alpha_t)}{\sqrt{\alpha_t}}s_\theta$$

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t + (1-\alpha_t)s_\theta)$$

.3 Task 1.3

Forward diffusion

$$x_t = \sqrt{\bar{\alpha}_t} X_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_\theta$$

Tweedie's formula

$$X_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t + (1-\bar{\alpha}_t)s_\theta)$$

$$x_t = \sqrt{\bar{\alpha}_t}(\frac{1}{\sqrt{\bar{\alpha}_t}}(x_t + (1-\bar{\alpha}_t)s_\theta)) + \sqrt{1-\bar{\alpha}_t}\epsilon_\theta$$

$$x_t = ((x_t + (1-\bar{\alpha}_t)s_\theta)) + \sqrt{1-\bar{\alpha}_t}\epsilon_\theta$$

$$-(1-\bar{\alpha}_t)s_\theta = \sqrt{1-\bar{\alpha}_t}\epsilon_\theta$$

$$s_\theta = -\frac{\sqrt{1-\bar{\alpha}_t}}{(1-\bar{\alpha}_t)}\epsilon_\theta$$

Prove

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t + (1-\alpha_t)s_\theta) \Leftrightarrow x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t + \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}})\epsilon_\theta)$$

$$(1-\alpha_t)s_\theta \Leftrightarrow -\frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta$$

$$-(1-\alpha_t)\frac{\sqrt{1-\bar{\alpha}_t}}{(1-\bar{\alpha}_t)}\epsilon_\theta \Leftrightarrow -\frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta$$

$$-\frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta \Leftrightarrow -\frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta$$