Solving Inverse Problems with Diffusion Model

Fangjun Zhou

Abstract—Inverse problems in computational imaging are often ill-posed. In this project, I explore the use of diffusion models as learned image priors for solving these problems. Using a pretrained diffusion model on the FFHQ dataset, I first study the forward and reverse diffusion processes and demonstrate unconditional image generation. I then apply three diffusion-based restoration methods, SDEdit, ScoreALD, and Diffusion Posterior Sampling (DPS), to the image inpainting and deconvolution task. The results show that diffusion models can generate realistic reconstructions. SDEdit exhibits a trade-off between realism and faithfulness, while ScoreALD and DPS achieve better reconstruction quality overall. Among the tested methods, DPS performs best in both visual quality and quantitative metrics, showing the benefit of more accurate posterior guidance.

Index Terms—Diffusion Model, Computational Imaging

1 Introduction

The inverse problems in computational imaging are often ill-posed, which means that the solution may not be unique or numerically stable. Two of the examples of such problems are image inpainting and image deconvolution. In image inpainting, the task is to fill in the missing pixels in an image. For a given image with missing pixels, there are infinitely many ways to fill in the missing pixels, and the solution may not be unique. In image deconvolution, the task is to recover a sharp image from a blurred image. For a given blurred image, the deblurred image are sensitive to the noise in the blurred image, and the solution may not be numerically stable.

In this study, I try to overcome the ill-posedness of the inverse problems by using a diffusion model as a prior. This limits the solution space to a distribution of natural images learned from a large dataset of images. Specifically, I use a pretrained diffusion model on the Flickr-Faces-HQ (FFHQ) dataset [1]. I started with generating images unconditionally using the backward diffusion process. Then, I solved the image inpainting and image deconvolution problems by applying the SDEdit, ScoreALD, and DPS algorithms proposed in [2], [3], and [4].

2 Related Work

Before the deep learning methods were being used, the image inpainting and image deconvolution problems were solved by using hand-crafted priors. For example, the total variation (TV) prior was used to solve the image deconvolution problem [5]. The TV prior encourages the solution to have piecewise constant regions, which is a common property of natural images. However, the TV prior may not be able to capture the detailed textures in natural images, and it may lead to over-smoothing of the solution. For the image inpainting problem, texture synthesis methods were used to fill in the missing pixels [6]. These methods fill in the missing pixels by copying and pasting patches from the known regions of the image. However, these methods can only fill in the missing pixels with textures that are present in the known regions of the image, and they may not be able to fill in the missing pixels with textures that are not present in the known regions.

3 Method

3.1 Forward Diffusion Process

In this project, I used the variance-preserving (VP) formulation of diffusion model defined in the DDPM paper [7]. During the forward diffusion process, the image x_t at time step t is generated by adding noise $z_t \sim \mathcal{N}(0, I)$ to the image x_{t-1} at time step $t-1$:

$$x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}z_t, \quad t = 1, 2, ..., T \tag{1}$$

Let $\alpha_t = 1 - \beta_t$. At step 1,

$$x_1 = \sqrt{1-\beta_1}x_0 + \sqrt{\beta_1}z_1 \tag{2}$$
$$= \sqrt{(\alpha_1)}x_0 + \sqrt{1-\alpha_1}z_1 \tag{3}$$

At step 2,

$$x_2 = \sqrt{1-\beta_2}x_1 + \sqrt{\beta_2}z_2 \tag{4}$$
$$= \sqrt{\alpha_1\alpha_2}x_0 + \sqrt{\alpha_2(1-\alpha_1)}z_1 + \sqrt{1-\alpha_2}z_2 \tag{5}$$
$$\tag{6}$$

At step t,

$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sum_{i=1}^{t}(\sqrt{1-\alpha_i}\prod_{j=i+1}^{t}\sqrt{\alpha_j})z_i \tag{7}$$

where $\bar{\alpha}_t = \prod_{i=1}^{t}\alpha_i$. Since $z_i \sim \mathcal{N}(0, I)$, the summation term is also a Gaussian noise with zero mean and variance $\sum_{i=1}^{t}(1-\alpha_i)\prod_{j=i+1}^{t}\alpha_j$.

$$\sum_{i=1}^{t}(1-\alpha_i)\prod_{j=i+1}^{t}\alpha_j = (1-\alpha_t)+\alpha_t(1-\alpha_{t-1})+... \quad (8)$$

$$+\prod_{i=2}^{t}\alpha_i(1-\alpha_1) \quad (9)$$

$$=1-\prod_{i=1}^{t}\alpha_i = 1-\bar{\alpha}_t \quad (10)$$

Therefore, the image x_t at time step t can be directly generated by adding noise $z \sim \mathcal{N}(0, I)$ to the original image x_0:

$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}z, \quad t = 1, 2, ..., T \quad (11)$$

3.2 Reverse Diffusion Process

The reverse diffusion process with a score predictor s_θ is defined as:

$$\hat{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t + (1-\bar{\alpha}_t)s_\theta(x_t, t)) \quad (12)$$

$$x_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\hat{x}_0 \quad (13)$$

Since $\hat{\alpha}_t = \alpha_t\bar{\alpha}_{t-1}$, we can rewrite the above equation as:

$$x_{t-1} = \frac{\alpha_t(1-\bar{\alpha}_{t-1})}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\hat{x}_0 \quad (14)$$

$$= \frac{1}{\sqrt{\alpha_t}}x_t + \frac{\alpha_t - 1}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}x_t \quad (15)$$

$$+ \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\hat{x}_0 \quad (16)$$

$$(17)$$

Since

$$\frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\hat{x}_0 = \frac{1-\alpha_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}x_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}s_\theta(x_t, t) \quad (18)$$

We have

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t + (1-\alpha_t)s_\theta(x_t, t)) \quad (19)$$

3.3 Noise Predictor and Score Predictor

As shown in Equation 11, the one step forward diffusion process can be directly computed with $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}z$. Since the Tweedie's formula states that the expected value of the original image x_0 given the noisy image x_t is

$$\mathbb{E}[x_0|x_t] = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t + (1-\bar{\alpha}_t)\delta_{x_t}\log p(x_t)) \quad (20)$$

where $\nabla_{x_t}\log p(x_t)$ is the score function of the noisy image x_t, we can rewrite the score function as:

$$\nabla_{x_t}\log p(x_t) = \frac{\sqrt{\bar{\alpha}_t}\mathbb{E}[x_0|x_t] - x_t}{1-\bar{\alpha}_t} \quad (21)$$

$$= \frac{\sqrt{\bar{\alpha}_t}\mathbb{E}[x_0|x_t] - \sqrt{\bar{\alpha}_t}x_0 - \sqrt{1-\bar{\alpha}_t}z}{1-\bar{\alpha}_t} \quad (22)$$

$$= -\frac{z}{\sqrt{1-\bar{\alpha}_t}} \quad (23)$$

So the reverse diffusion process with a score predictor s_θ can be rewritten with a noise predictor ϵ_θ as:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)) \quad (24)$$

3.4 Unconditional Image Generation

To generate images unconditionally, we can start with a random noise image $x_T \sim \mathcal{N}(0, I)$ at time step T, and then apply the reverse diffusion process iteratively to generate the image at time step $t-1$ from the image at time step t until we reach time step 0. The generated image is the image at time step 0.

3.5 Solving Inverse Problems

The inverse problems can be formulated as a posterior sampling problem. Given an observed image $y = Ax$ and a forward model A, we want to sample the original image x from the posterior distribution $p(x|y) = \frac{p(y|x)p(x)}{p(y)}$, where $p(y|x)$ is the likelihood function defined by the forward model, and $p(x)$ is the prior distribution defined by the diffusion model. The SDEdit, ScoreALD, and DPS algorithms are different methods to sample from the posterior distribution.

3.5.1 SDEdit

The SDEdit [2] algorithm is a simple method to approximate the sampling from the posterior distribution by adding noise to the observed image and then applying the reverse diffusion process. Specifically, given an observed image y, we can add noise to the observed image to get a noisy image $x_t = \sqrt{\bar{\alpha}_t}y + \sqrt{1-\bar{\alpha}_t}z$, where $z \sim \mathcal{N}(0, I)$. Then, we can apply the reverse diffusion process to generate the image at time step $t-1$ from the image at time step t until we reach time step 0. If we add a large amount of noise to the observed image, the generated image will be more diverse but may not be close to the observed image. If we add a small amount of noise to the observed image, the generated image will be less diverse but may be closer to the observed image. A special case of the SDEdit is that if $t = T$ (i.e., we add full noise to the observed image), the image generation process will be the same as the unconditional image generation.

3.5.2 ScoreALD

The ScoreALD [3] algorithm is a method to sample from the posterior distribution by guiding the reverse diffusion process with the gradient of the log-likelihood function. Specifically, given an observed image $y = Ax$ and the image formation model A, we can compute the gradient of the log-likelihood function $\nabla_{x_t}\log p(y|x)$, and then add

this gradient to the score predictor during the reverse diffusion process. This will guide the reverse diffusion process towards generating images that are more likely to produce the observed image under the forward model.

One challenge of the ScoreALD is that we don't know the exact value of the original image x during the reverse diffusion process, so we will assume that $x_t \approx x$ when computing the gradient of the log-likelihood function. So for each time step t, we have

$$\hat{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t + (1-\bar{\alpha}_t)s_\theta(x_t, t)) \tag{25}$$

$$x'_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\hat{x}_0 \tag{26}$$

$$x_{t-1} = x'_{t-1} - \zeta\nabla_{x_t}\log p(y|x_t) \tag{27}$$

This assumption may not be accurate, especially at the early steps of the reverse diffusion process when the generated image is still very noisy.

3.5.3 DPS

The DPS [4] algorithm introduces a more accurate approximation to the gradient by assuming $\hat{x}_0 \approx x$ instead of $x_t \approx x$. So for each time step t, we have

$$\hat{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t + (1-\bar{\alpha}_t)s_\theta(x_t, t)) \tag{28}$$

$$x'_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\hat{x}_0 \tag{29}$$

$$x_{t-1} = x'_{t-1} - \zeta\nabla_{x_t}\log p(y|\hat{x}_0) \tag{30}$$

4 Experiments

4.1 Image Denoising

In the image denoising experiment, I added Gaussian noise to the original image to get x_t from x and apply the reverse diffusion process to get the predicted clear image $\hat{x}_0$. Figure 1 shows the results for different time steps t.

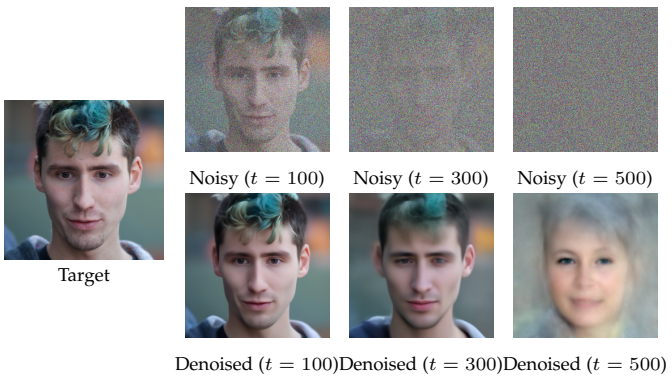

Fig. 1: Left: target image. Right: top row shows noisy images at different diffusion timesteps, and bottom row shows the corresponding denoised outputs.

I measured the PSNR and LPIPS between the target image and the denoised image for different time steps t. The results are shown in Table 1.

TABLE 1: Image denoising metrics at different diffusion timesteps.

Time Step t	PSNR	LPIPS
100	31.5402	0.1069
300	21.1101	0.3218
500	15.7029	0.6890

The results show that the denoised image at time step $t = 100$ has the highest PSNR and lowest LPIPS. As the time step increases, the PSNR decreases and the LPIPS increases. This shows that as we add more noise to the original image, it becomes harder for the reverse diffusion process to recover the original image.

4.2 Unconditional Image Generation

As described in Section 3.4, the unconditional image generation can be achieved by starting with a random noise image at time step T and applying the reverse diffusion process iteratively until we reach time step 0. Figure 2 shows three examples of the generated images.

Fig. 2: Unconditional image generation results.

Since the diffusion model is trained on the FFHQ dataset, the generated images are sampled from human face distribution.

4.3 SDEdit

I applied the SDEdit algorithm on the image inpainting and deconvolution problems with different time steps t. The results are shown in Figure 3.

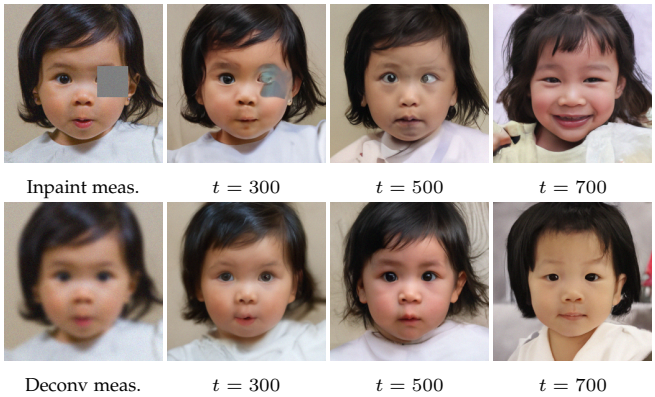

Fig. 3: Top row: inpainting measurement and reconstructions at different timesteps. Bottom row: deconvolution measurement and reconstructions at different timesteps.

I measured the PSNR and LPIPS between the target image and the reconstructed image for different time steps t. The results are shown in Table 2.

TABLE 2: SDEdit quantitative results for deconvolution and inpainting.

Task	Time Step t	PSNR	LPIPS
Deconv	300	22.3842	0.3984
Deconv	500	19.9994	0.3865
Deconv	700	15.4317	0.4585
Inpaint	300	22.5026	0.3180
Inpaint	500	20.2576	0.3810
Inpaint	700	15.4789	0.4446

As shown in the results, the reconstructed images at time step $t = 300$ have the highest PSNR and lowest LPIPS for both deconvolution and inpainting tasks. As the time step increases, the PSNR decreases and the LPIPS increases. Qualitative results also shows that the reconstructed images at a smaller time step tend to produce more artifacts, while the reconstructed images at a larger time step tend to be more realistic but less similar to the target image.

This phenomenon was described in the original SDEdit paper [2] as a trade-off between realism and faithfulness. As Meng et al. [2] explained, using a smaller time step (i.e., adding less noise to the observed image) will make the reverse diffusion process more faithful to the observed image, but it may produce more artifacts. Using a larger time step (i.e., adding more noise to the observed image) will make the reverse diffusion process more realistic, but it may be less faithful to the observed image.

5 ScoreALD and DPS

The result of ScoreALD algorithm on the image inpainting and deconvolution problems with annealing factor $\zeta = [1, 1.5]$ are shown in Figure 4. The result of DPS algorithm with annealing factor $\zeta = [0.1, 1.0]$ are shown in Figure 5.

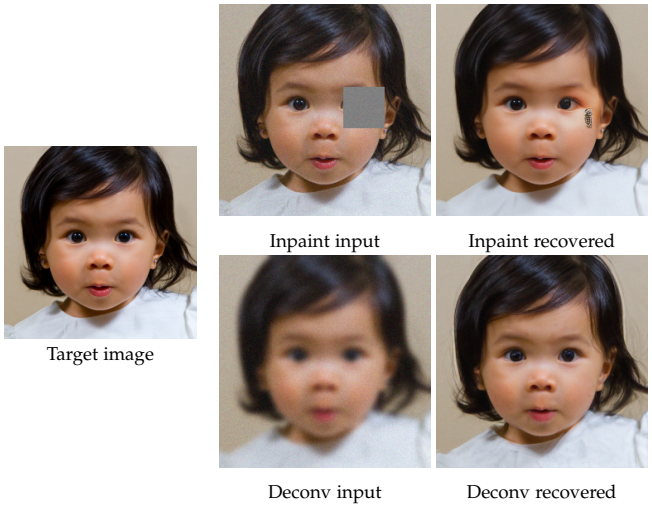

Fig. 4: ScoreALD results. Left: target image. Right: input and recovered images for inpainting (top) and deconvolution (bottom).

The PSNR and LPIPS between the target image and the reconstructed image for both inpainting and deconvolution tasks are shown in Table 3.

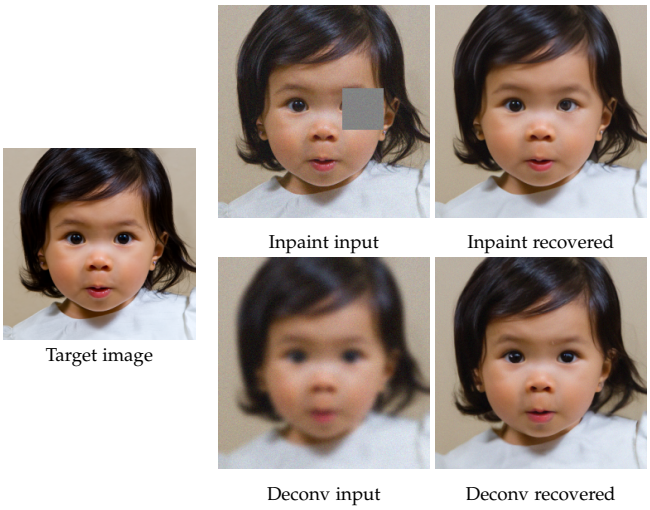

Fig. 5: DPS results. Left: target image. Right: input and recovered images for inpainting (top) and deconvolution (bottom).

TABLE 3: Quantitative results for ScoreALD and DPS on inpainting and deconvolution.

Method	Task	PSNR	LPIPS
ScoreALD	Inpaint	28.3267	0.2241
ScoreALD	Deconv	26.2732	0.2420
DPS	Inpaint	29.4600	0.2363
DPS	Deconv	27.1175	0.2627

As shown in the results, the reconstructed images from both ScoreALD and DPS algorithms have higher PSNR and lower LPIPS than the reconstructed images from the SDEdit algorithm. And DPS performs better than ScoreALD visually and in terms of PSNR. This is because the ScoreALD algorithm uses a less accurate approximation to the gradient of the log-likelihood function, which may lead to suboptimal guidance during the reverse diffusion process. The DPS algorithm uses a more accurate approximation to the gradient, which can provide better guidance and lead to better reconstruction results.

5.1 Hybrid Image

On top of the Inpainting and Deconvolution tasks, I also tried to solve a hybrid image problem by applying the DPS algorithm. The hybrid image assumes the image formation model is the same as the convolution and try to solve $y = Ax$ for two different blurred images y_1 and y_2.

Specifically, I applied the convolution filter for two different faces and alternating the reverse diffusion process for the two images. The result is shown in Figure 6.

6 Conclusion

In this project, I studied how diffusion models can be used as image priors to solve ill-posed inverse problems. I implemented unconditional image generation and applied diffusion-based posterior sampling methods to inpainting and deconvolution.

Among the posterior sampling methods I evaluated, SDEdit demonstrates the trade-off between faithfulness and

Input 1 Input 2 Hybrid result

Fig. 6: Hybrid image experiment using DPS.

realism, while ScoreALD and DPS produce higher-quality reconstructions both visually and quantitatively. In particular, DPS achieved the best performance in my experiments, suggesting that more accurate likelihood guidance leads to better posterior sampling results. Overall, these results highlight the effectiveness of diffusion models for solving inverse problems.

REFERENCES

[1] T. Karras, S. Laine, and T. Aila, "A Style-Based Generator Architecture for Generative Adversarial Networks," Mar. 2019.

[2] 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," Jan. 2022.

[3] A. Jalal, M. Arvinte, G. Daras, E. Price, A. G. Dimakis, and J. I. Tamir, "Robust Compressed Sensing MRI with Deep Generative Priors," Dec. 2021.

[4] H. Chung, J. Kim, M. T. Mccann, M. L. Klasky, and J. C. Ye, "Diffusion Posterior Sampling for General Noisy Inverse Problems," May 2024.

[5] D. Perrone and P. Favaro, "Total Variation Blind Deconvolution: The Devil Is in the Details," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*. Columbus, OH, USA: IEEE, Jun. 2014, pp. 2909–2916.

[6] A. Efros and T. Leung, "Texture synthesis by non-parametric sampling," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Kerkyra, Greece: IEEE, 1999, pp. 1033–1038 vol.2.

[7] J. Ho, A. Jain, and P. Abbeel, "Denoising Diffusion Probabilistic Models," Dec. 2020.