Imaging With Diffusion Model Prior

Scott Milner

Abstract—Diffusion models have recently gained prominence as a critical component of many image processing pipelines. In this work, we review and explore the basic capabilities of a pre-trained image diffusion model when used as as a prior. We explore methods for unconditional generation, deconvolution, and inpainting. To further refine results, we examing multiple methods for conditioning from related works, including ScoreALD and DPS methods.

Index Terms—Computational Photography, Diffusion Models, Stanford EE367

1 Introduction

Correctly identifying and dealing with noise is a critical component of signal processing methods, and image processing is no different. We can define an image formation model as the following:

$$\mathbf{b} = \mathbf{A}\mathbf{x} + \eta \tag{1}$$

where $\mathbf{A}$ is our measurement method, $\mathbf{x}$ is our ground truth signal, η is our signal noise, and $\mathbf{b}$ is our measured/observed signal. Accurately identifying the value of η is critical to reconstructing the the true value of $\mathbf{x}$ given $\mathbf{b}$.

Many methods have been proposed and adopted over the years; diffusion models have recently gained traction as a prominent tool for identifying and assisting in processing noise. Notably, diffusion models' differentiable properties lend themselves to *inverse* imaging uses, such as unconditioned image generation, image inpainting, and image deconvolution.

Herein we will first discuss basic uses of diffusion models such as one-shot denoising and unconditioned generation. Afterwards, we will explore several prior works in the field of image reconstructing (specifically inpainting and deconvolution) whose methods make use of diffusion models, namely Score-Distillation Editing (SDEdit) [1], Score Annealed Langevin Dynamics (ScoreALD) [2], and Diffusion Posterior Sampling (DPS) [3].

2 Related Work

The basis for diffusion-based denoising is covered in depth by Ho et. al in *Denoising Diffusion Probabilistic Models* [4]. Therein, Ho lays out the case for diffusion models over other methods like Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs), demonstrating that unlike in attempts from previous works, diffusion models are capable of producing high-quality results. For example, while GANs produce good results, they are more difficult to train and are difficult to generalize to other tasks, such as the inpainting we will be using a diffusion model for. Ho et. al demonstrate unconditional generation with their model, and their work has been built upon by many other authors in subsequent years, including the the methods listed above that we will be exploring here [1] [2] [3].

• *S. Milner is with the Department of Computer Science, Stanford, CA, 94305*

3 Proposed Method

We will begin by presenting the structure of the methods we will be using.

3.1 Diffusion Models

To use our diffusion model we define two processes: a forward process and a backward process.

3.1.1 *Forward Process*

The forward process "noises" an image by iteratively adding noise to it, subject to a noise schedule "beta" schedule (a monotonic, usually linear schedule $\beta_t \in [0, 1]$, where $t \in \mathbb{Z}^+ | t < T$ and T is the number of steps in the schedule). We define the forward process as follows:

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\mathbf{z}_{t-1} \tag{2}$$

where $\mathbf{x}_t$ represents the noised measurement after time t and each $\mathbf{z}_t \sim \mathcal{N}(0, I)$ represents a Gaussian random variable. By working out the first couple of terms we can derive an explicit forward process based only on the initial (noiseless) measurement (using $\alpha_t = 1 - \beta_t$):

$$\begin{aligned}
\mathbf{x}_1 &= \sqrt{\alpha_1}\mathbf{x}_0 + \sqrt{1-\alpha_1}\mathbf{z}_0 \\
\mathbf{x}_2 &= \sqrt{\alpha_2}\mathbf{x}_1 + \sqrt{1-\alpha_2}\mathbf{z}_1 \\
&= \sqrt{\alpha_2\alpha_1}\mathbf{x}_0 + \sqrt{\alpha_2(1-\alpha_1)}\mathbf{z}_0 + \sqrt{1-\alpha_2}\mathbf{z}_1 \\
&= \sqrt{\alpha_2\alpha_1}\mathbf{x}_0 + \sqrt{\alpha_2(1-\alpha_1) + 1 - \alpha_2}\mathbf{z} \\
&= \sqrt{\alpha_2\alpha_1}\mathbf{x}_0 + \sqrt{1-\alpha_2\alpha_1}\mathbf{z} \\
\mathbf{x}_3 &= \sqrt{\alpha_3}\mathbf{x}_2 + \sqrt{1-\alpha_3}\mathbf{z}_2 \\
&= \sqrt{\alpha_3}\left(\sqrt{\alpha_2\alpha_1}\mathbf{x}_0 + \sqrt{1-\alpha_2\alpha_1}\mathbf{z}_2\right) + \sqrt{1-\alpha_3}\mathbf{z}_3 \\
&= \sqrt{\alpha_3\alpha_2\alpha_1}\mathbf{x}_0 + \sqrt{\alpha_3 - \alpha_3\alpha_2\alpha_1}\mathbf{z}_2 + \sqrt{1-\alpha_3}\mathbf{z}_3 \\
&= \sqrt{\alpha_3\alpha_2\alpha_1}\mathbf{x}_0 + \sqrt{1-\alpha_3\alpha_2\alpha_1}\mathbf{z}
\end{aligned}$$

After generalizing $\forall t$ by substituting $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\mathbf{z} \tag{3}$$

Fig. 1. One-shot denoising after $t = 10, 100, 500$ steps of forward noising

becomes our final model for the forward process, allowing us to immediately noise a measurement to the appropriate step along the beta schedule.

3.1.2 Backward Process

The DDPM paper [4] uses the following model as the backwards process:

$$\hat{\mathbf{x}}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + (1-\bar{\alpha}_t)\mathbf{s}_\theta(\mathbf{x}_t, t)) \tag{4}$$

where $\hat{\mathbf{x}}_0$ is the predicted original measurement and $\mathbf{s}_\theta(\mathbf{x}_t, t)$ is the score-predicting function, trained to match $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$. The score-predicting function $\mathbf{s}_\theta$ is interchangeable with the noise-predicting function ϵ_θ, which is described by this relation:

$$\mathbf{s}_\theta(\mathbf{x}_t, t) = -\frac{\epsilon_\theta(\mathbf{x}_t, t)}{\sqrt{1-\bar{\alpha}_t}} \tag{5}$$

Using this method, we are able to do "one-shot denoising": estimating the noise present in the image and subtracting it off to get an an estimation of the original measurement.

The score-predicting function has nicer properties

While one-shot denoising is a good start, we can get much better results if we take an incremental approach: denoising a small amount and then requerying the model to figure out the next step, repeating until we reach the end of the beta schedule. The DDPM paper gives the following equation for an incremental denoising approach.

$$\mathbf{x}_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\hat{\mathbf{x}}_0 \tag{6}$$

By combining (5) with (6) we can get an incremental denoising step that only relies on $\mathbf{x}_t$, not $\hat{\mathbf{x}}_0$:

Fig. 2. Three examples of unconditional generation with a model trained on the FFHQ-256 dataset

$$\begin{aligned}
\mathbf{x}_{t-1} &= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t \\
&\quad + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t} \cdot \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + (1-\bar{\alpha}_t)\mathbf{s}_\theta(\mathbf{x}_t, t)) \\
&= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{\sqrt{\bar{\alpha}_t}(1-\bar{\alpha}_t)}\mathbf{x}_t \\
&\quad + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)(1-\bar{\alpha}_t)\mathbf{s}_\theta(\mathbf{x}_t, t)}{\sqrt{\bar{\alpha}_t}(1-\bar{\alpha}_t)} \\
&= \frac{\alpha_t(1-\bar{\alpha}_{t-1}) + 1 - \alpha_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}\mathbf{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\mathbf{s}_\theta(\mathbf{x}_t, t) \\
&= \frac{1-\bar{\alpha}_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}\mathbf{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\mathbf{s}_\theta(\mathbf{x}_t, t)
\end{aligned}$$

Yielding

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}(\mathbf{x}_t + (1-\alpha_t)\mathbf{s}_\theta(\mathbf{x}_t, t)) \tag{7}$$

as our final backward process for denoising a measurement.

3.2 Unconditional Generation

Now that we have a backwards process defined, we can perform unconditional generation, that is, starting with pure noise and slowly step backwards towards $\mathbf{x}_0$, allowing an undirected signal to emerge from the noise. The model we used is trained on the Flickr Faces High-Quality (FFHQ-256) dataset, giving it a particular affinity for inputs with human faces. Thus all unconditioned generation outputs from this model resemble faces.

3.3 Image Reconstruction

We next explore methods of image reconstruction using diffusion models. In this paper we explore two kinds of reconstruction problems: inpainting and deconvolution. The first of these, inpainting, involves masking off a part of the image completely and using the model to reconstruct the missing portion of the image. The mask can be either a contiguous portion of the image or a random selection of pixels across the entire image. The second kind of reconstruction, deconvolution, involves blurring an image to remove high-frequency detail. We then seek to reconstruct the missing detail.

3.3.1 Score-Distillation Editing

The first reconstruction method we will look at is Score-Distillation Editing (SDEdit) as proposed by Meng et. al. [1] The SDEdit is the naïve approach to reconstruction: we apply the forward process to our corrupted measurement (**b**) until we reach a predetermined step along the beta schedule, then trace our steps back using the backward process. As the original corrupted measurement is unlikely to be in the distribution of the model, this can sometimes successfully reconstruct the corrupted measurements.

3.3.2 Score Annealed Langevin Dynamics

The next reconstruction method we will explore is Score Annealed Langevin Dynamics (ScoreALD) [2]. Unlike SDEdit, ScoreALD works by adding conditioning to the standard unconditional generation process. Instead of beginning with a partially noised image, ScoreALD begins with a fully noised image and conditions the backwards step to resolve towards the measurement ($\mathbf{y}$). The conditioning term for ScoreALD is based off of $\mathbf{x}_t$ the current image, $\mathbf{y}$ the measurement, and γ_t, a hyperparameter that scales the strength of the conditioning factor over time.

$$\frac{1}{\sigma^2 + \gamma_t^2} \nabla_{\mathbf{x}_t} ||\mathcal{A}(\mathbf{x}_t) - \mathbf{y}|| \tag{8}$$

Thus the full ScoreALD algorithm is

$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \ldots, 1$ **do**
  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
  $\hat{\mathbf{x}}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t + (1 - \bar{\alpha}_t)\boldsymbol{s}_\theta(x_t, t))$
  $\boldsymbol{x}_{t-1} = \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t}\boldsymbol{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1 - \alpha_t)}{1 - \bar{\alpha}_t}\hat{\boldsymbol{x}}_0 + \sqrt{1 - \alpha_t}\boldsymbol{z}$
  $\boldsymbol{x}_{t-1} = \boldsymbol{x}_{t-1} - \frac{1}{2(\sigma^2 + \gamma_t^2)} \nabla_{\boldsymbol{x}_t} \left\| \mathscr{A}(\boldsymbol{x}_t) - \boldsymbol{y} \right\|^2$
end for
return $\mathbf{x}_0$

3.3.3 Diffusion Posterior Sampling

The final reconstruction method we will explore is Diffusion Posterior Sampling (DPS), as described by Chung et. al. [3] Similar to ScoreALD, DPS begins with a fully noised image and performs conditional generation by conditioning the backwards diffusion process. Notably, DPS conditioning is based off of the current estimate of the ground truth $\hat{\mathbf{x}}_0$ instead of the current image $\mathbf{x}_t$. The DPS conditioning term is also described by a hyperparameter scalar $\zeta \in (0, 1]$:

$$\frac{\zeta}{|| \nabla_{\mathbf{x}_t} ||\mathcal{A}(\hat{\mathbf{x}}_0) - \mathbf{y}||^2 ||} \nabla_{\mathbf{x}_t} ||\mathcal{A}(\hat{\mathbf{x}}_0) - \mathbf{y}||^2 \tag{9}$$

Thus the full DPS algorithm is

TABLE 1
Image reconstruction Results

Method	Deconvolution		Inpainting	
	PSNR	LPIPS	PSNR	LPIPS
SDEdit ($t = 0.10$)	25.12	0.2687	22.93	0.1589
SDEdit ($t = 0.25$)	24.02	0.1807	20.87	0.2045
SDEdit ($t = 0.50$)	21.16	0.1916	16.45	0.2590
ScoreALD	25.33	0.1050	24.38	0.1278
DPS	26.36	0.0909	30.82	0.0512

$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \ldots, 1$ **do**
  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
  $\hat{\mathbf{x}}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t + (1 - \bar{\alpha}_t)\boldsymbol{s}_\theta(x_t, t))$
  $\boldsymbol{x}_{t-1} = \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t}\boldsymbol{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1 - \alpha_t)}{1 - \bar{\alpha}_t}\hat{\boldsymbol{x}}_0 + \sqrt{1 - \alpha_t}\boldsymbol{z}$
  $\boldsymbol{x}_{t-1} = \boldsymbol{x}'_{t-1} - \zeta_t \nabla_{\boldsymbol{x}_t} \left\| \mathscr{A}(\hat{\boldsymbol{x}}_0) - \boldsymbol{y} \right\|^2$
end for
return $\mathbf{x}_0$

---

where $\zeta_t = \frac{\zeta}{|| \nabla_{\mathbf{x}_t} ||\mathcal{A}(\hat{\mathbf{x}}_0) - \mathbf{y}||^2 ||}$.

4 Experimental Results

Figure 3 shows our results for the reconstruction tasks (inpainting and deconvolution) across all three methods detailed above: SDEdit, ScoreALD, and DPS. For SDEdit, we show three differnt t-values to demonstrate the affects of various degrees of noising has on the effectiveness and fidelity of the algorithm.

4.1 Quantitative Analysis

To analyze of the accuracy of these algorithms, we will use two metrics: Peak Signal-to-Noise-Ratio (PSNR) and Learned Perceptual Image Patch Similarity (LPIPS). The first of these, PSNR, describes how well the algorithm does at eliminating random noise and effectively isolating our ground-truth signal. While mathematically precise and good for describing signal strength, LPIPS can be a more useful metric for describing accuracy in reconstructing image features that a human eye might be sensitive to. LPIPS uses its own model to parameterize the ground truth and reconstructed image, then uses the difference between model parameters to score the reconstruction. LPIPS provides a metric more similar to what a human brain might pick up on with regard to feature similarity. A higher PSNR indicates a more faithful reconstruction and a lower LPIPS score indicates less perceptual feature loss.

Figure 3 is annotated with the PSNR and LPIPS scores; they are also included in Table 1 for convenient comparison.

In comparing t-values for SDEdit, we see that low values of t result in better PSNR scores but worse LPIPS scores. There is a sweet spot around $t \approx 0.25$ with more optimal scores. ScoreALD and DPS both significantly outperform

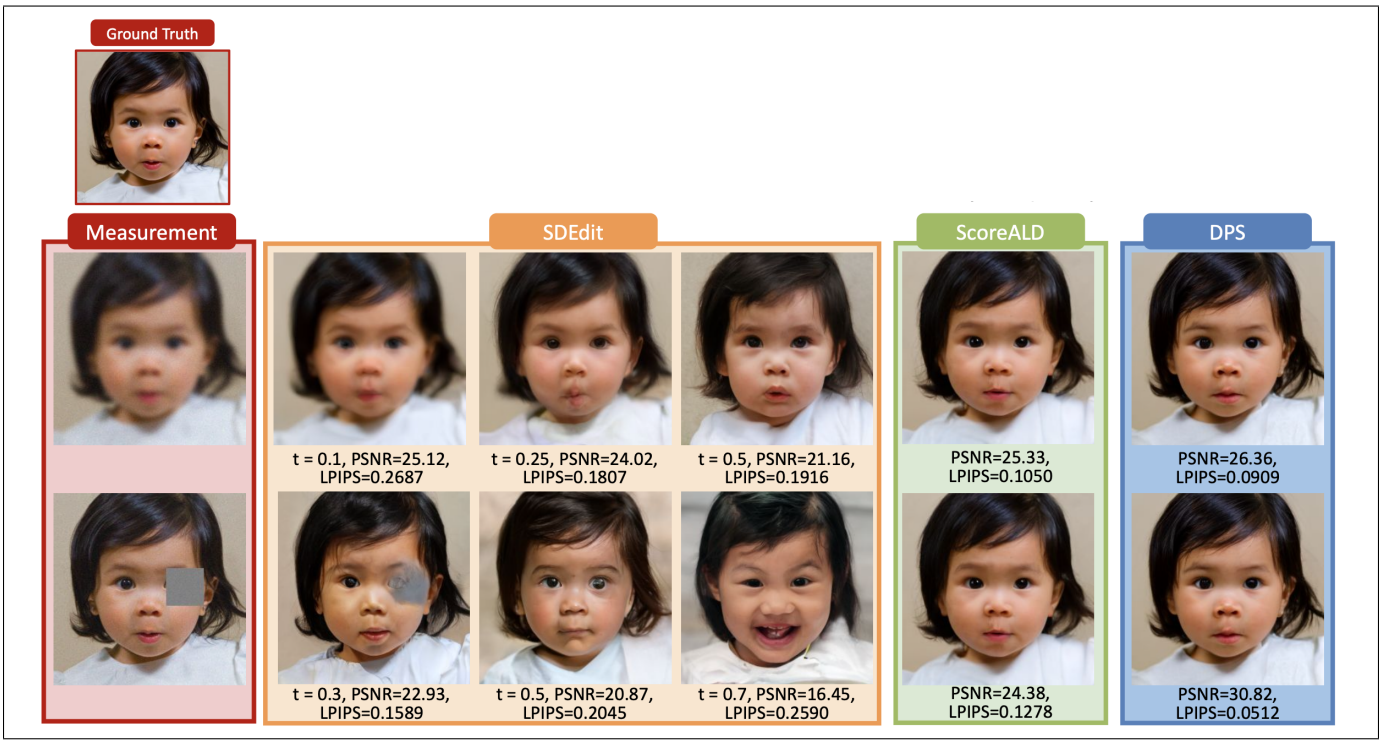

Fig. 3. Results from three different reconstruction methods on inpainting and deconvolution tasks.

SDEdit, and DPS performs better than ScoreALD on all metrics.

4.2 Qualitative Analysis

Overall, SDEdit has a difficult time preserving the fidelity of the original image. Logically, this makes sense as we add noise on top of the original measured (corrupted) signal and then never attempt to guide the image back towards the signal during the diffusion process; it is not unlikely for the signal to "stray" as we diffuse back towards a low-noise image. We observe that when the t-value is too low, the diffusion process does not have enough time to reconstruct the missing information, resulting in the gray blob around the eye in the inpainting example and in the blurry features in the deconvolution example. However, as the t-value increases, the fidelity to the ground truth rapidly decreases and once we reach a point where SDEdit fully reconstructs missing information, the face appears significantly different (see when $t = 0.5$ for our example image).

ScoreALD and DPS perform significantly better, their results have high similarity to the ground truth. Both successfully reconstruct the details of the face, as well as the shape of the bangs and avoid hallucinating new detail in the background. DPS performs the best, this is particularly noticable when observing the high frequency detail on the shirt, the W-shape curve of the upper lip, and the fine hair detail in the lower right.

5 Conclusion

In this project, we examined applications of the diffusion model prior in computer imaging. We looked at uses for a diffusion model that can predict the noise or score of the noise in the image, and how it integrates into processing pipelines for unconditional generation, denoising, deconvolution, and image inpainting. We explored three different methods for image reconstruction with a diffusion model prior: SDEdit, ScoreALD, and DPS, observing that DPS gives the best results, followed by ScoreALD, and then SDEdit.

References

[1] 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," 2022. [Online]. Available: https://arxiv.org/abs/2108.01073

[2] A. Jalal, M. Arvinte, G. Daras, E. Price, A. G. Dimakis, and J. I. Tamir, "Robust compressed sensing mri with deep generative priors," 2021. [Online]. Available: https://arxiv.org/abs/2108.01368

[3] H. Chung, J. Kim, M. T. Mccann, M. L. Klasky, and J. C. Ye, "Diffusion posterior sampling for general noisy inverse problems," 2024. [Online]. Available: https://arxiv.org/abs/2209.14687

[4] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," 2020. [Online]. Available: https://arxiv.org/abs/2006.11239

APPENDIX

Additional Derivations

.1 Proof 1

Proof of (3)

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\mathbf{z}_{t-1}, \;\; t = 1, 2, ..., T$$

Plug in first couple of iterations (substituting $\alpha_t = 1 - \beta_t$):

$$\begin{aligned}
\mathbf{x}_1 &= \sqrt{\alpha_1}\mathbf{x}_0 + \sqrt{1-\alpha_1}\mathbf{z}_0 \\
\mathbf{x}_2 &= \sqrt{\alpha_2}\mathbf{x}_1 + \sqrt{1-\alpha_2}\mathbf{z}_1 \\
&. = \sqrt{\alpha_2}\left(\sqrt{\alpha_1}\mathbf{x}_0 + \sqrt{1-\alpha_1}\mathbf{z}_0\right) + \sqrt{1-\alpha_2}\mathbf{z}_1 \\
&= \sqrt{\alpha_2\alpha_1}\mathbf{x}_0 + \sqrt{\alpha_2(1-\alpha_1)}\mathbf{z}_0 + \sqrt{1-\alpha_2}\mathbf{z}_1 \\
&= \sqrt{\alpha_2\alpha_1}\mathbf{x}_0 + \sqrt{\alpha_2(1-\alpha_1) + 1 - \alpha_2}\mathbf{z} \\
&= \sqrt{\alpha_2\alpha_1}\mathbf{x}_0 + \sqrt{\alpha_2 - \alpha_2\alpha_1 + 1 - \alpha_2}\mathbf{z} \\
&= \sqrt{\alpha_2\alpha_1}\mathbf{x}_0 + \sqrt{1-\alpha_2\alpha_1}\mathbf{z}
\end{aligned}$$

$$\begin{aligned}
\mathbf{x}_3 &= \sqrt{\alpha_3}\mathbf{x}_2 + \sqrt{1-\alpha_3}\mathbf{z}_2 \\
&= \sqrt{\alpha_3}\left(\sqrt{\alpha_2\alpha_1}\mathbf{x}_0 + \sqrt{1-\alpha_2\alpha_1}\mathbf{z}_2\right) + \sqrt{1-\alpha_3}\mathbf{z}_3 \\
&= \sqrt{\alpha_3\alpha_2\alpha_1}\mathbf{x}_0 + \sqrt{\alpha_3 - \alpha_3\alpha_2\alpha_1}\mathbf{z}_2 + \sqrt{1-\alpha_3}\mathbf{z}_3 \\
&= \sqrt{\alpha_3\alpha_2\alpha_1}\mathbf{x}_0 + \sqrt{1-\alpha_3\alpha_2\alpha_1}\mathbf{z}
\end{aligned}$$

Generalize to $\mathbf{x}_t$

$$\begin{aligned}
\mathbf{x}_t &= \sqrt{\prod_{i=1}^{t}\alpha_i}\mathbf{x}_0 + \sqrt{1-\prod_{i=1}^{t}\alpha_i}\mathbf{z} \\
&= \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\mathbf{z}
\end{aligned}$$

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\mathbf{z}, \;\; \alpha_t = 1 - \beta_t, \bar{\alpha}_t = \prod_{i=1}^{t}\alpha_i \quad (10)$$

.2 Proof 2

Proof of (7). Given:

$$\hat{\mathbf{x}}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + (1-\bar{\alpha}_t)\mathbf{s}_\theta(\mathbf{x}_t, t))$$

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

$$\begin{aligned}
\mathbf{x}_{t-1} &= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\hat{\mathbf{x}}_0 \\
&= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t \\
&\quad + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t} \cdot \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + (1-\bar{\alpha}_t)\mathbf{s}_\theta(\mathbf{x}_t, t)) \\
&= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)(\mathbf{x}_t + (1-\bar{\alpha}_t)\mathbf{s}_\theta(\mathbf{x}_t, t))}{\sqrt{\bar{\alpha}_t}(1-\bar{\alpha}_t)} \\
&= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{\sqrt{\bar{\alpha}_t}(1-\bar{\alpha}_t)}\mathbf{x}_t \\
&\quad + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)(1-\bar{\alpha}_t)\mathbf{s}_\theta(\mathbf{x}_t, t)}{\sqrt{\bar{\alpha}_t}(1-\bar{\alpha}_t)} \\
&= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}\mathbf{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\mathbf{s}_\theta(\mathbf{x}_t, t) \\
&= \frac{\alpha_t(1-\bar{\alpha}_{t-1}) + 1 - \alpha_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}\mathbf{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\mathbf{s}_\theta(\mathbf{x}_t, t) \\
&= \frac{\alpha_t - \bar{\alpha}_t + 1 - \alpha_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}\mathbf{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\mathbf{s}_\theta(\mathbf{x}_t, t) \\
&= \frac{1-\bar{\alpha}_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}\mathbf{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\mathbf{s}_\theta(\mathbf{x}_t, t) \\
&= \frac{1}{\sqrt{\alpha_t}}\mathbf{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}}\mathbf{s}_\theta(\mathbf{x}_t, t) \\
&= \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t + (1-\alpha_t)\mathbf{s}_\theta(\mathbf{x}_t, t)\right)
\end{aligned}$$

.3 Proof 3

Proof of (5). Given:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_\phi(\mathbf{x}_t, t)$$

Substituting in Tweedie's formula we get:

$$\begin{aligned}
\mathbf{x}_0 &= \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + (1-\bar{\alpha}_t)\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t)) && (11)\\
&= \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + (1-\bar{\alpha}_t)\mathbf{s}_\theta(\mathbf{x}_t, t)) && (12)
\end{aligned}$$

We can then substitute this into (3) (substituting $\mathbf{z}$ with $\epsilon_\phi(\mathbf{x}_t, t)$ and simplify:

$$\begin{aligned}
\mathbf{x}_t &= \sqrt{\bar{\alpha}_t} \cdot \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + (1-\bar{\alpha}_t)\mathbf{s}_\theta(\mathbf{x}_t, t)) + \sqrt{1-\bar{\alpha}_t}\epsilon_\phi(\mathbf{x}_t, && (13)\\
\mathbf{x}_t &= \mathbf{x}_t + (1-\bar{\alpha}_t)\mathbf{s}_\theta(\mathbf{x}_t, t) + \sqrt{1-\bar{\alpha}_t}\epsilon_\phi(\mathbf{x}_t, t) && (14)\\
(1-\bar{\alpha}_t)\mathbf{s}_\theta(\mathbf{x}_t, t) &= -\sqrt{1-\bar{\alpha}_t}\epsilon_\phi(\mathbf{x}_t, t) && (15)\\
\mathbf{s}_\theta(\mathbf{x}_t, t) &= -\frac{\sqrt{1-\bar{\alpha}_t}}{1-\bar{\alpha}_t}\epsilon_\phi(\mathbf{x}_t, t) && (16)\\
&= -\frac{\epsilon_\phi(\mathbf{x}_t, t)}{\sqrt{1-\bar{\alpha}_t}} && (17)
\end{aligned}$$

APPENDIX

Additional Results

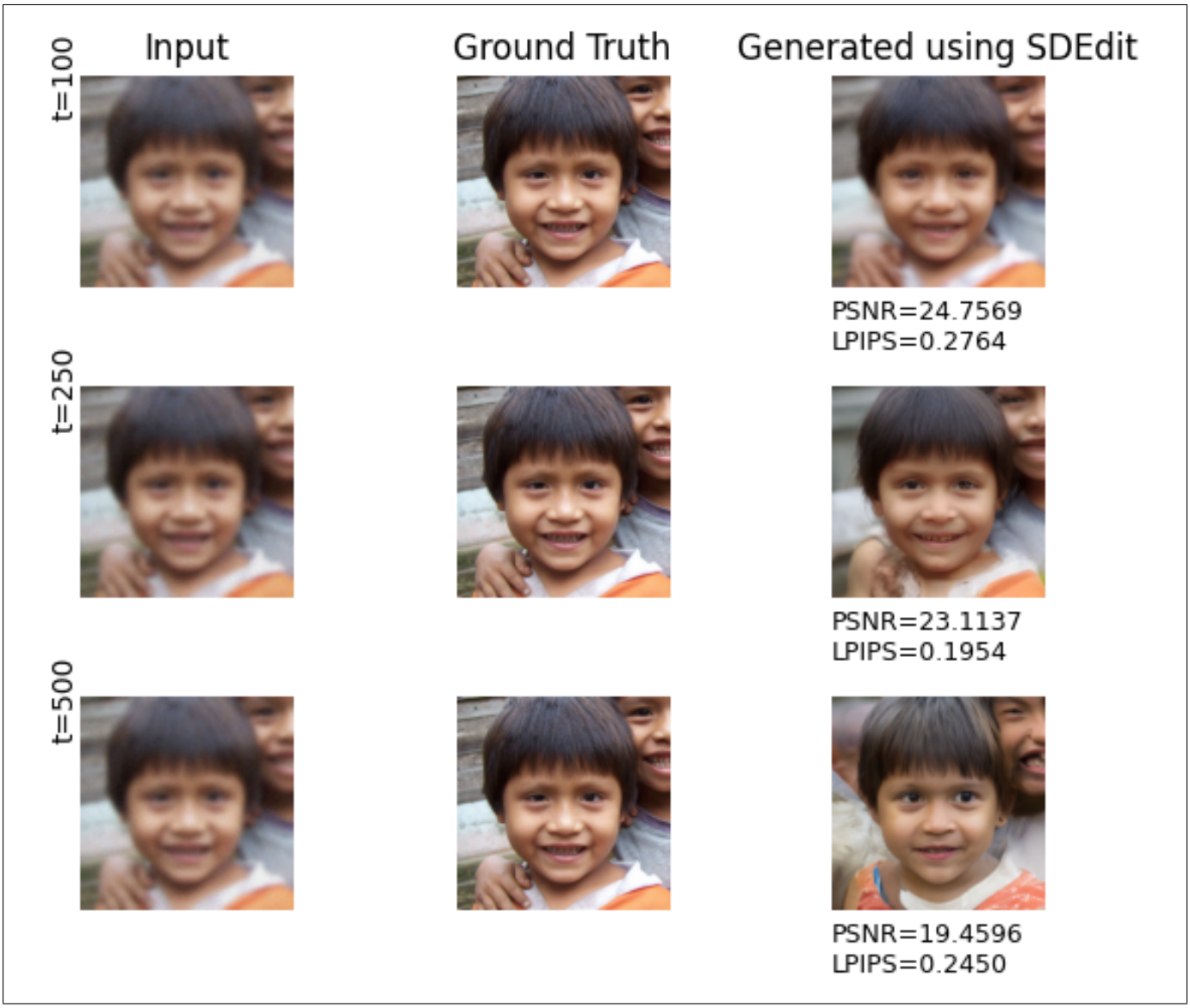

Fig. 4. Additional SDEdit results