Diffusion Model Exploration

Ryan Hurley

Abstract - This paper covers the usage of diffusion models for natural image generation and for solving inverse problems such as deconvolution and inpainting, Overall it shows that diffusion models can produce accurate results numerically and visually in both tasks. The methods used in this paper are SDEdit, ScoreALD and DPS. This outperforms other conventional methods for image deconvolution such as Adam, HQS, and ADMM which are highly dependent on the choice of prior and regulators. Diffusion models have a forward process to procedurally add noise to an image and a reverse process which is learned from the forward process to remove noise in each step.

Index Terms - Computational Photography

---

1 Introduction

DIFFUSION models can be applied to many cases such as image generation, editing, text-to-images, video generation and more. This paper explores applications of diffusion models in image generation and solving inverse problems such as deconvolution and inpainting. Diffusion models have a forward and reverse process. The models are typically trained on large image datasets and hallucinate new images according to user parameters.

The second section will cover similar work to diffusion models such as Adam, HQS and ADMM in solving deconvolution and inpainting. Next the new methodology and results. Finally a conclusion and references.

The diffusion models used in this paper were trained with FFHQ-256 dataset. This is the Flickr-Faces-HQ Dataset and includes 256x256 pixel images. Image generation should result in images that could appear to fit into that dataset.

2 Related Work

Similar work includes using Adam, HQS and DnCNNs for deconvolution. This work was done on test images and the resulting PSNRs were compared. The results presented in increasing quality on the PSNR scale. The overall PSNR difference is small though, less than half a dB over the three methods.

Adaptive Moment Estimation (Adam) for deconvolution with the total variation (TV) norm. The TV norm is a prior that attempts to assess natural images. Typically natural images are often smoother and the TV norm represents this smoothness factor. The resulting image had much less noise than the input but a lot of quality was lost and the results are visually smooth.

Half Quadratic Splitting (HQS) with the TV norm performed similarly to Adam. This method alternates updating the x and z variables which are used to separate the terms. X is often called the data fidelity term and z is the regularizer. HQS allows for easy swapping of the regularizer term to test different methodologies to describe images. In this case the TV norm was used.

HQS was used also with a denoising convolution neural network (DnCNN). This no longer uses a direct equation such as the TV norm as a regularizer but now an entire neural network. This had the most performance increase over other methods currently but still produces images that have quality to be desired.

3 Methodology

The next five sections cover each method that was used in solving inverse problems. All methods are run in a Jupiter notebook and execute Python code with the models stored on Google Drive. The fundamentals of each method were derived and implemented as needed. The models have one thousand steps in the diffusion model and the noise is Gaussian noise.

3.1 Image Denoising

The forward process for image denoising is procedurally each step adding small amounts of noise to an image. This happens many times for a given dataset. In the FFHQ-256 dataset this includes a variety of human headshot photos. These get progressively noisier until after one thousand steps they become complete noise. A neural network is trained in this forward process to learn how the noise is applied.

The reverse process is starting from a noisy image and slowly removing noise one step at a time until a clean image is generated. These images are not directly the trained dataset but variations and permutations of data near it. These models use primarily gradient descent to look for a suitable solution in the final image.

The forward pass can also be applied in a single step to get a noisy image after a known number of samples. This can be done by the following equation.

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

This calculates the image at step t, x_t from the original image x_0 and the amount of noise added in each step alpha bar. The denoiser is the rearranged version of the formula.

$$x_0 = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\epsilon}{\sqrt{\bar{\alpha}}}$$

The score is calculated by

$$score = -\frac{\epsilon}{\sqrt{1-\bar{\alpha}_t}}$$

Epsilon is the noise predicted from the neural network trained in the forward process. This method produces images that have been denoised from an input noisy image.

3.2 Image Generation

The same method of diffusion models is applied to the image generation problem. The fundamentals are the same as the image denoising task. The same denoiser is used on the noisy image x_t to predict the entire clean image x_0 hat. Once the clean image is predicted a single step of noise is removed.

Bayes' rule of posterior distribution is applied to solving this process resulting in the following formulation of the posterior mean.

$$\mu(x_t x_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} x_t$$

This is simplified into the posterior mean coefficients multiplied by x_o and x_t respectively. The purpose of this variable is to manipulate how the clean image and the noisy image are weighted. This means the first few steps which have mostly noise in the image rely on the second term which has the noisy image. When the image is closer to being generated the posterior mean is more depending on the predicted clean image or the first term.

The input is just a completely noisy Gaussian image. Inputting the same exact image into the model will return the same exact result every time. In this case the input noise is randomized to return different results each time. To process an image the diffusion model does the denoising, clean image prediction and posterior mean calculations in order a thousand times to produce a new image.

3.3 Solving Inverse Problems with SDEdit

Score-based Diffusion Editing is a method to solve inverse problems by removing most of the noise and re-adding noise in the middle and removing it to generate new results. This and future methods rely heavily on the methods used in the image generation section and are variations of the diffusion models used there.

To perform inpainting the image is masked and only a small section of the partially generated image has more noise applied onto it. This changes how the gradient descent works as it is making the image more noisy. Then when the image has more noise added the reverse process is run. The forward process if fully applied, then the reverse process is started. Partially through the reverse process in a masked section the forward process is partially applied again. After, the reverse process is completed resulting in editing of the generated image. The masking follows the following formula.

$$x_{t-1} = M \odot y + (1 - M) \odot x_{t-1}$$

In deconvolution SDEdit uses a scaling factor on calculating the next step denoised image by a data fidelity term. The amount of editing can be modified by

manipulating the noise applied at step t_0. Large values of t_0 allow almost an entirely new image to be generated while small values of t_0 produce small modifications to the image. The goal of SDEdit is to start from a partially clear and partially noisy image to allow for tweaking of the image before it is completed.

3.4 Solving Inverse Problems with ScoreALD

Score-based Annealed Langevin Dynamics (ScoreALD) function similar to the previous methods but it has a special anneal factor. The Langevin Dynamics are as follows.

$$x_{k+1} = x_k + \frac{\eta}{2}\nabla_x log(x_k) + \sqrt{\eta} z_k$$

The score function is used here and represented by the gradient of the log. The step size is eta. Anneal is required as at the start of the gradient descent large steps and optimal to close the distance to an optimal solution quickly. Then as the noise levels decrease the anneal factor changes to decrease the step size. The new update is defined as

$$x_{t-1} = x_{t-1} - \frac{1}{2(\sigma^2+\gamma^2)}\nabla_{x_t}||A(x_t) - y||^2$$

The anneal factor is multiplied by gradient likelihood to allow for optimal results. The anneal factor scales depending on the task. Lower values are used for deconvolution and slightly higher values are used for inpainting. The ScoreALD method uses both the prior of the denoise network and a likelihood function scaled by the anneal factor.

3.5 Solving Inverse Problems with DPS

Diffusion Posterior Sampling (DPS) also has many similarities to the mentioned methods. However the key difference with DPS is another alternation to the likelihood gradient function. The new update function is defined as.

$$x_{t-1} = x'_{t-1} - \zeta_t \nabla_{x_t}||A(\widehat{x_0}) - y||^2$$

This methodology requires no new training for the net and works very well in these inverse problems. The images it outputs are natural looking when derived by the function above.

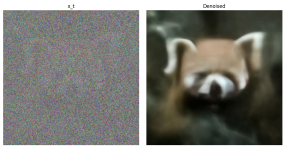
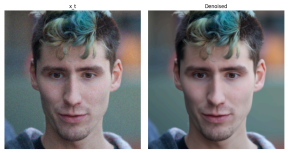
4. Results

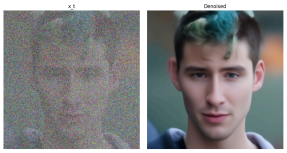
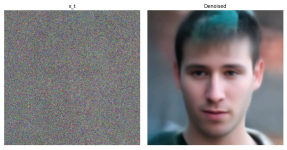
The outputs from each method are compared visually and numerically. PSNR of each output image is calculated which compares the pixel mean squared error. 30+ dB is considered good, 25-30 dB is average and under 25 dB is poor image quality. An ideal PSNR would be infinite if two images are compared. PSNR does not exactly align well with human vision and is more of a numerical way to compare images.

LPIPS is a more advanced method to determine image quality and is itself a deep neural network to compare two images. LPIPS of 0 would be a perfect match. Less than 0.1 is good quality, 0.1-0.2 is average and over 0.3 is poor. The LPIPS score is much closer to representing image quality perceived by the human visual system.

4.1 Image Denoising

The following images are two separate tests with different levels of noise. The less noise initially added the closer the diffusion model is to predicting the expected result. It is important to note that even with a seemingly almost fully noisy image it is still able to generate some features of the inputted image.

Input / Output Image	T	PSNR	LPIPS
	10	39.3580888872214 4	0.014977613463997841
	100	28.8883180815957 84	0.18470847606658936
	250	25.7070624830354 5	0.5586408972740173
	500	21.8786295345516 6	0.6766138672828674
	10	41.5954365073847 1	0.011879548443044281

	100	32.5918896077119 8	0.09160100668668747
	250	28.2404316817273 3	0.17724062502384186
	500	23.3416652933420 18	0.35763430595539795

4.2 Image Generation

The following images were generated based on the FFHQ-256 dataset. The resulting images are in similar form to other images in the dataset. The results are not perfect visually and have some slightly unrealistic features however.

4.3 Solving Inverse Problems with SDEdit

The following results are images generated with SDEdit. SDEdit did not perform out of the three methods used but was able to produce realistic natural results with the right manipulation of parameters. The first image is the ground truth, the second is the input into the inpainting problem and the third is the input into the deconvolution problem.

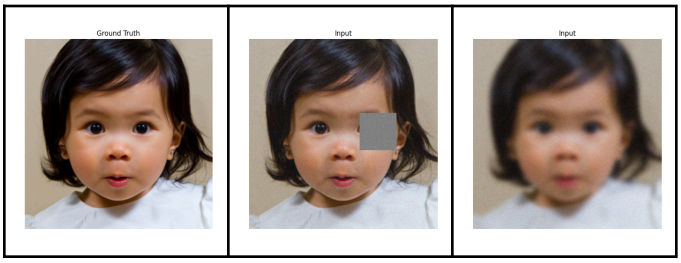


Image	Method	T	PSNR	LPIPS
	Inpaint	450	21.2075489594221	0.1711816042661667
	Inpaint	500	20.9492108213796 23	0.1788708120584488
	Inpaint	550	19.3924938351905 57	0.22140538692474365

	Inpaint	800	13.1760516161777439	0.3871976137161255
	Deconv	450	20.9180145301332 63	0.2001009881496429 4
	Deconv	500	21.2190725586627 06	0.1712341308593 75
	Deconv	550	18.7737129614957 47	0.2255257070064544 7
	Deconv	800	13.1177651305477 72	0.4178024530410766 6

4.4 Solving Inverse Problems with ScoreALD

ScoreALD had decent performance improvements over SDEdit numerically. Visually the images had closer to intended features but had some variations with the image colors. The same inputs as the SDEdit method apply to the ScoreALD as well.

Image	Method	PSNR	LPIPS
	Inpaint	23.0551709380 6918	0.1225691437721 2524
	Deconv	22.0193856883 67095	0.1546180099248 886

4.5 Solving Inverse Problems with DPS

DPS had the highest image quality visually and numerically. DPS also had the same inputs as ScoreALD and SDEdit.

Image	Method	PSNR	LPIPS
	Inpaint	29.28148190026606	0.04990855231881142
	Deconv	27.036686678518187	0.08382998406887054

5 Conclusion

DPS had the best performance visually and numerically but it also took a little extra time to run the network. All of the methods produced visually appealing results but some required more tuning and configuration to do so. Diffusion models proved to excel in solving inverse problems such as deconvolution and inpainting. These models are also able to be configured to output randomized natural images.

Limitations included the dataset trained and what images it includes. Every single person is not included in this image set so this limits its capabilities to generate a variety of images. The networks are also very computer resource intensive and powerful computers are required for computation. Google Colab was used for their GPU cloud network. However this was not always accessible during the duration of the project due to demand.

Future work may include a larger dataset with more options or the capability to influence the image generation with text prompts. The models could also be used to solve other inverse problems.

6 References

[1] Boyd, S., et al. (2011). Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers.

[2] Chung, H., Kim, J., Mccann, M. T., Klasky, Ml L., and Ye, J. C. (2023). Diffusion posterior sampling for general noisy inverse problems. In ICLR

[3] Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. In NeurIPS.

[4] Jalal, A., Arvintre, M., Daras, G., Price, E., Dimakis, A. G., and Tamir, J. (2021). Robust compressed sensing mri with deep generative priors.

[5] Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization.

[6] Meng, C., He, Y., Song, Y., Song, J., Wu, J., Zhu, J.-Y., and Ermon, S. (2022). Sdedit: Guided image synthesis and editing with stochastic differential equations.

[7] Venkatakrishnan, S. V., et al. (2013). Introducing Plug-and-Play Priors for Model-Based Iterative Reconstruction.