Score-Based Generative Modeling through Reverse Diffusion Monte Carlo

Ethan Vertal
Statistics and Machine Learning
Carnegie Mellon University
Pittsburgh, PA 15213
evertal@andrew.cmu.edu

Mudita Sai
Statistics and Machine Learning
Carnegie Mellon University
Pittsburgh, PA 15213
mksai@andrew.cmu.edu

Supriya Shingade
Statistics and Machine Learning
Carnegie Mellon University
Pittsburgh, PA 15213
sshingad@andrew.cmu.edu

1 Introduction

We will perform class-conditional image generation using ImageNet data with the intention of comparing experimental results of these new methods to the standard Denoising Diffusion Probabilistic Models or shortly, DDPM (Ho et. al., 2020) and standard score-based models. We propose a novel implementation of a Reverse Diffusion Monte Carlo (rdMC) sampler and re-implementation of a DDPM and Score-based Generative diffusion model via stochastic differential equations. Our hope is that the Monte Carlo sampling-based approximation methods will achieve near or better performance on this task, as measured by Fréchet Inception Distance (FID). We will also compare this new model with state-of-the-art GANs (Po et. al., 2023), VAEs and energy-based models.

2 Dataset and Task

We will be using a 5,200 image subset of the ImageNet Large Scale Visual Recognition Challenge (ILVRC) dataset. We first began with using a dataset subsetted from a single class, cats, to evaluate the model's performance on a singular class-level basis. If time and memory permits, we will extend our dataset to span over multiple classes. The reason we subset the dataset is because we are aiming to experiment with various Monte Carlo sampling techniques in place of the traditional parameterization of the reverse diffusion process. That being said, we will be focusing on an image generation task of cats.

3 Related Work

There exist four broad principal categories of generative models (Song et. al., 2021). The first of which are GANs that perform in a min-max adversarial fashion. Secondly, Variational Autoencoders (VAEs), autoencoders, and normalizing flows belong to the category rooted in likelihood-based approaches. The third category encompasses energy-based modeling, wherein the distribution is modeled as an energy function that is subsequently normalized. The final category includes score-based matching where the score of the energy based model is learned as a neural net. Current state-of-the-art diffusion models, such as DDPMs, leverage denoising score matching with annealed Langevin dynamics and a parameterized Markov chain trained via variational inference to generate samples approximating the underlying data distribution (Ho et al., 2020). This falls under our second category of generative models. However, it is noteworthy that these various generative models more or less make use of similar objective functions. Current literature in diffusion models showcase three equivalent objectives that optimize the model by learning a neural network to 1. predict the original image x_0, 2. predict the source noise ϵ_0 and 3. compute the score of the image, $\nabla \log p(x_t)$ (Luo, 2022). Nonetheless, a crucial element in reversing the learned diffusion model hinges upon the efficiency of convergence from any complex distribution to a normal distribution. Usually the probability distribution of our data, $p(x)$ can be defined as follows:

$$p_{\boldsymbol{\theta}}(x) = \frac{1}{Z_{\boldsymbol{\theta}}} e^{-f_{\boldsymbol{\theta}}(x)}$$

where $f_{\boldsymbol{\theta}}(x)$ is the energy function which can be modeled by a neural network and $Z_{\boldsymbol{\theta}}$ is a normalizing constant to ensure that $\int p_{\boldsymbol{\theta}}(x)dx = 1$. $Z_{\boldsymbol{\theta}}$ is intractable especially for complex distributions but score-based generative modeling completely avoids the need to learn this normalizing constant. By using a neural net to learn the score function $s_{\boldsymbol{\theta}}(x)$ we bypass the need to use $Z_{\boldsymbol{\theta}}$. The score function is defined as follows:

$$\begin{aligned}
\nabla_x \log p_{\boldsymbol{\theta}}(x) &= \nabla_x \log(\frac{1}{Z_{\boldsymbol{\theta}}} e^{-f_{\boldsymbol{\theta}}(x)}) \\
&= \nabla_x \log \frac{1}{Z_{\boldsymbol{\theta}}} + \nabla_x \log e^{-f_{\boldsymbol{\theta}}(x)} \\
&= -\nabla_x f_{\boldsymbol{\theta}}(x) \\
&\approx s_{\boldsymbol{\theta}}(x)
\end{aligned}$$

where $s_{\boldsymbol{\theta}}(x)$ is the score function computed from an arbitrary time-dependent neural network that minimizes the Fisher Divergence as defined:

$$\frac{1}{2}\mathbb{E}_{p(x)}\left[\|s_{\boldsymbol{\theta}}(x) - \nabla \log p(x)\|_2^2\right] \tag{1}$$

But the Fisher divergence cannot be directly computed, since the score of the data distribution $\nabla \log p(x)$ is unknown. Fortunately, score matching eliminates the data score using integration by parts and so our new score matching objective (which is equivalent to the above objective function) can be defined as follows:

$$\mathbb{E}_{p(x)}\left[tr(s_{\boldsymbol{\theta}}^2(x)) + \frac{1}{2}\|s_{\boldsymbol{\theta}}(x)\|_2^2\right] + C \tag{2}$$

where minimizing the above eq. only requires knowledge of $s_{\boldsymbol{\theta}}(x)$. Once the score function is learned, samples can be generated starting from a random point and iteratively following the score through a process known as Langevin Dynamics.

$$\mathbf{x}_{i+1} = \mathbf{x_i} + c\nabla \log p(\mathbf{x}_i) + \sqrt{2c}\epsilon \tag{3}$$

The usual process is to randomly sample from the from the prior distribution with some additive noise $\epsilon \sim \mathcal{N}(\epsilon; \mathbf{0}, \mathbf{I})$ which prevents the model from deterministically converging to the mode and getting trapped in a local minima. However, Langevin Dynamics is prone to a multitude of problems such as when x is a low dimension manifold in a high dimensional space or when the score is not accurate in low density regions of the distribution or getting stuck a local minimas. The solution to these problems all involve adding multiple levels of Gaussian noise which allows the data sample to not be confined to a local minima/low dimensional manifold which gives rise to a method called Annealed Langevin Dynamics (Song and Ernon 2020, [1]). To train the neural network $\mathbf{s}_\theta(\mathbf{x}, \sigma)$ we will use a noise conditional score network (Song and Ernon [2]). Thus the resulting neural net is time-dependent and noise-conditional which is as follows:

$$\arg\min_\theta \sum_{t=1}^{T} \lambda(t)\mathbf{E}_{p_{\sigma_t}}(x_t)[\|\mathbf{s}_\theta(\mathbf{x}, t) - \nabla \log p_{\sigma_t}(\mathbf{x_t})\|] \tag{4}$$

where $\lambda(t)$ is a positive weighting function that conditions on noise level t. To generalize this process to infinite number of time steps (and thereby accounting for an infinite perturbations) Stochastic Differential Equations come into play. The images are sampled simply by reversing the SDE. In a similar vein, the reverse diffusion Monte Carlo (rdMC) algorithm which also samples data from an unnormalized distribution is designed in such a way to incorporate score-based modeling (Dong et. al., 2024). But the rdMC diverges from conventional Langevin MCMC techniques in utilizing a reverse Stochastic Differential Equation (SDE), specifically the Ornstein-Uhlenbeck (OU) process, which is also able to represent a time-homogeneous continuous-time Markov process observed at non-uniformly sampled discrete times (Santos, Lin, 2024).

4 Approach

Since the forward process of a diffusion model has no learnable parameters and can be computed efficiently in one pass with the help of the reparametrization trick, our focus will be on the more interesting reverse process which in essence is the denoising process and can be implemented in a multidude of ways.

4.1 Denoising Diffusion Probablistic Model

Our absolute baseline model is the traditional Denoising Diffusion Probablistic Model, presented in the landmark paper by Ho et. al. in 2020. This approach involves training the model via minimization of the negative log-likelihood of $p_\theta(\mathbf{x})$. In other words, we train it via Maximum Likelihood Estimation, but since $p_\theta(\mathbf{x})$ is generally intractable, we maximize the Evidence Lower Bound (ELBO):

$$\log p(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T} \mid \mathbf{x}_0)}\right]. \tag{5}$$

Much of this approach is discussed in Section 3. However, it is important to recognize that the training process in this approach involves sampling from a reparameterization of a Gaussian distribution, passing that sample through a model that learns how to apply noise to these samples to replicate $\mathcal{N}(0, \mathbf{I})$, and taking the L2 loss of the model output with a random sample from said distribution. Then, using this newly trained noise model, we input noise and transform the model output via the algorithm found in Ho et. al., 2020, where they use a UNet to model the learned noise from the training process.

4.2 Reverse-Diffusion Monte Carlo

Rather than reparameterizing the Gaussian distribution as in the DDPM method, we are interested in using Monte Carlo Methods to perform the reverse diffusion process. And as such, we will try to implement the new reverse diffusion Monte Carlo (RdMC) sampling technique. RdMC recognizes that the reverse process is a specific Stochastic Differential Equation called an Ornstein-Uhlenbeck process, which also models the noise from a standard normal to the target distribution.

$$d\mathbf{x}_t = -\mathbf{x}_t dt + \sqrt{2} dB_t, \quad \mathbf{x}_0 \, p_0 \propto e^{f_*} \tag{6}$$

where B_t is a Brownian motion term and f_* represents the negative log-likelihood of our target distribution $p_0 = p_*$. We use the fact that this is an OU process to form the reverse equation as

$$\tilde{x}_{t+s} = e^s \tilde{x}_t + (e^s - 1)v_k + \mathcal{N}(0, (e^{2s} - 1)I_d), \text{ where } v_k \text{ is a variable for } \nabla_x \ln p_{T-t}$$

So what we have now is that,

$$v_k \approx \nabla_x \ln p_{T-t}(x) = \mathbb{E}_{x_0 \sim q_{T-t}}\left[\frac{e^{-(T-t)}x_0 - x}{(1 - e^{-2(T-t)})}\right],$$

$$q_{T-t}(x_0|x) \propto \exp\left(-f_*(x_0) - \frac{\|x - e^{-(T-t)}x_0\|^2}{2(1 - e^{-2(T-t)})}\right).$$

And now, using a sampling method similar to the Euler-Maruyama method we plan to implement as a simpler model, our algorithm from Huang et. al. is as shown in Algorithm 1. Our main goal here is still to estimate $v_k = \nabla_\mathbf{x} \log p(\mathbf{x}_t)$, but instead of directly approximating it with a UNet as in Huang et. al., we approximate v_k via taking the sample mean of n samples using the Unadjusted Langevin Algorithm to estimate its mean. Using the notation Huang et. al. and algorithms introduced in their 2024 paper that this approach is based off of, let $\{x_k^{(i)}\}_{i=1}^n$ represent samples taken in the ULA inner loop, provided in Algorithm 2 below.

Algorithm 1 RDMC: reverse diffusion Monte Carlo

```
1: Input: Initial particle $\tilde{x}_0$ sampled from $\tilde{p}_0$, Terminal time $T$, Step size $\eta$, $\eta'$, Sample size $n$.
2: for $k = 0$ to $\left\lfloor \frac{T}{\eta} \right\rfloor - 1$ do
3:     Set $v_k = 0$;
4:     Create $n$ Monte Carlo samples to estimate
           $$v_k \approx \mathbb{E}_{x \sim q_{T-t}} \left[ \frac{\tilde{x}_{kn} - e^{-(T-k\eta)} x}{(1 - e^{-2(T-k\eta)})} \right]$$
5:     $\tilde{x}_{(k+1)\eta} = e^{\eta} \tilde{x}_{kn} + (e^{\eta} - 1) v_k + \xi$ where $\xi$ is sampled from $\mathcal{N}(0, (e^{2\eta} - 1) I_d)$.
6: end for
7: Return: $\tilde{x}_{\lceil T/\eta \rceil \eta}$.
```

Algorithm 2 ULA inner-loop for the $q_t(\cdot|x)$ sampler (Step 4 of Algorithm 1)

```
1: Input: Condition $x$ and time $t$, Sample size $n$, Initial particles $\{x_0^i\}_{i=1}^n$, Iters $K$, Step size $\eta$.
2: for $k = 1$ to $K$ do
3:     for $i = 1$ to $n$ do
4:         $x_k^i = x_{k-1}^i - \eta \left( \nabla f_*(x_{k-1}^i) + \frac{e^{-t}\left(e^{-t} x_{k-1}^i - x\right)}{1 - e^{-2t}} \right) + \sqrt{2\eta} \xi_k,$ where $\xi_k \sim \mathcal{N}(0, I_d)$
5:     end for
6: end for
7: Return: $\{x_K^i\}_{i=1}^n$.
```

5 Experiments

As for experiments with the Score-Based method and rdMC, we will run the models for 20,000 steps with the goal of sampling from our 5,200 cat image dataset. As of now, we have completed training and sampling of these images for our DDPM model. The results can be seen in Figures 1 and 2 below. We started with a learning rate of 1e-3 for 10,000 training steps and then reduced it to 1e-5 for the next 5,000 steps, and reduced it further to 1e-8 for the last 5,000 steps (hence the three loss plots from Weights and Biases).

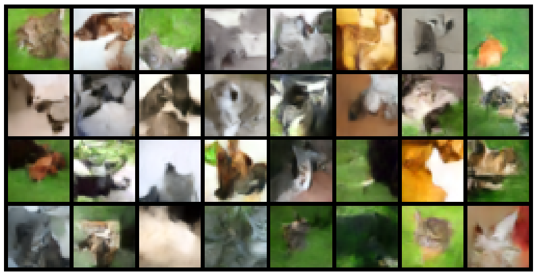
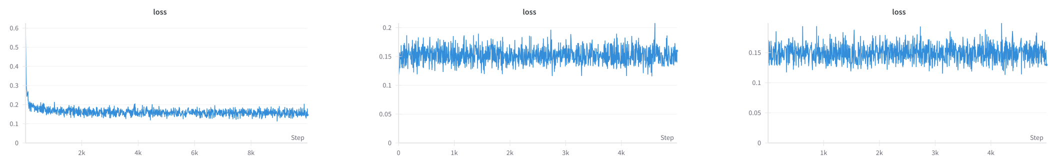


Figure 1: Samples after 20k training steps from DDPM



(a) LR: 1e-3 (b) LR: 1e-5 (c) LR: 1e-8

Figure 2: L1 Loss for all 20k steps, will include more graphs or more lines for other models

Below is a, currently empty, table containing the FID scores for all three models. This will serve as a comparison, but since we have done little experimentation so far we have no results for this yet.

Model	FID
DDPM	-
Score-Based GM	-
RDMC	-

Table 1: Comparison of models based on FID score

6 Plan

We plan on first creating the score based generative model (4/25), implementing reverse diffusion Monte Carlo (4/28), comparing generated images across models (4/29), creating poster (4/30), completing the report (5/2) and finally presenting our results (5/2). We also plan on continuing to work together in person as our main form of completing the programming portions as these methods are generally conceptually difficult for us to understand. Our general goal is to get these milestones done as a group, working together collaboratively such that we all have an equal hand in each milestone. It is important for us to contribute to each step since these all provide valuable state-of-the-art research experience, given our belief that rdMC has not been openly implemented before.

Bibliography

Ho, J., Jain, A., & Abbeel, P. (2020). Denoising Diffusion Probabilistic Models (arXiv:2006.11239). arXiv. `http://arxiv.org/abs/2006.11239`

Huang, X., Dong, H., Hao, Y., Ma, Y.-A., & Zhang, T. (2024). Reverse Diffusion Monte Carlo (arXiv:2307.02037). arXiv. `http://arxiv.org/abs/2307.02037`

Luo, C. (2022). Understanding Diffusion Models: A Unified Perspective (arXiv:2208.11970). arXiv. `http://arxiv.org/abs/2208.11970`

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2021). Score-Based Generative Modeling through Stochastic Differential Equations (arXiv:2011.13456). arXiv. `http://arxiv.org/abs/2011.13456`

Santos, J., Lin, Y. (2023). Using Ornstein–Uhlenbeck Process to understand Denoising Diffusion Probabilistic Model and its Noise Schedules (arXiv:2311.1767). arXiv. `http://arxiv.org/abs/2311.1767`

[1] Song, Y., Ermon S., (2020). Generative Modeling by Estimating Gradients of the Data Distribution (arXiv:1907.05600). arXiv. `http://arxiv.org/abs/1907.05600`

[2] Song, Y., Ermon S., (2020). Improved Techniques for Training Score-Based Generative Models (arXiv:2006.09011). arXiv. `http://arxiv.org/abs/2006.09011`

Po1, R., Yifan1, W., Golyanik2, V., Aberman, K., Barron, J. T., Bermano A., Chan, E., Dekel T., Holynski A., Kanazawa, A., Liu K., Mildenhall, B., Nießner M., Ommer B., Theobalt C., Wonka P., Wetzstein G. (2023). State of the Art on Diffusion Models for Visual Computing (arXiv:2310.07204). arXiv.http://arxiv.org/abs/2310.07204