Conditional Generative Diffusion with Langevin Dynamics

Vasily Ilin, Jonathan Friesen

Abstract

For our project, we wanted to engage in a general literature review on the mathematics of diffusion generative modeling, culminating in a particularly mathematically-rich version formulation, based on a reversal of the Ornstein-Uhlenbeck process using the time-dependent vector field $\nabla_x \log p_t(x)$ called the *score*. We look into the process of conditional generation in such models and implement the algorithms discussed to perform conditional generation over the MNIST dataset, exploring basic properties of such generation.

1 Introduction

In 2019 Song and Ermon [7, 8] started a small revolution in the world of sampling. They proposed a conceptually simple architecture able to sample from extremely high-dimensional ($d = 256 \times 256$) distributions, only needing access to an abundance of samples. Because the training is unsupervised (or semi-supervised in case of conditional generation, to be described later) it is easy to assemble a large dataset of images to train on.

In this paper we explore a particularly mathematically-rich version of diffusion generative models, based on a reversal of the Ornstein-Uhlenbeck process using the time-dependent vector field $\nabla_x \log p_t(x)$ called the *score*.

In section 2 we present the theoretical underpinnings of diffusion generative models, giving context to and explanations of Song and Ermon's proposal. In section 3 we introduce conditional generation, which makes generative models actually useful. Finally, in section 4 we perform several experiments on the MNIST dataset.

2 Background

2.1 Problem Formulation

When we have some data distributed $\mathcal{D} = \{x_i\}_{i=1}^n \overset{i.i.d.}{\sim} p$, where $p(x)$ is supposed to be unknown, we may naturally want to find the best estimate of this distribution as informed by the data (this is after all, the fundamental question of Statistical Inference). In the context of deep learning, this will inevitably entail using DNNs to learn $p_\theta(x)$, parameterized by weights θ. With this learned distribution, we could then sample $x_{new} \sim p_\theta$. Such a process is called Generative Modeling, and it has found immense empirical success in image, text and audio domains over the past few years.

There have been myriad of Deep Learning technologies trying to tackle this problem over the past few decades, but the origin of Diffusion Models can be traced to Sohl-Dickstein Et. al 2015 [6], with what the authors called Diffusion Probabilistic Models (DPM). Inspired by models in non-equilibrium statistics physics, these authors proposed a process of iteratively corrupting the structure of input data via a forward diffusion process Markov Chain with Gaussian noise transition kernels, and then learning the transition kernels for a reversal diffusion process in which structure is restored to this data.

This paper set the concept of corruption and reversal as the basic paradigm for Diffusion Models, but there is some departure from DPM and most contemporary Diffusion Models: namely, the former models are trained with a Maximum-Likelihood based loss (i.e. $\nabla_\theta \log p_\theta(x)$ via Bayesian Variational Bounds), while the latter is generally trained on a score-based loss, where we perform GD on $\nabla_x \log p_\theta(x)$. In our exposition of Diffusion Models, we shall focus on this latter formulation, briefly motivating why it should be preferable to a Maximum-Likelihood based model, and heuristically explaining the mathematics that explain its efficacy.

2.2 Score-Based Models

Likelihood based models are ultimately trying to maximize $\log p_\theta(x_1, ..., x_n) = \sum_{i=1}^{n} p_\theta(x_i)$, the probability of the data in $\mathcal{D}$ occurring. Thus at some point, the models under consideration need to be constrained to proper probability distributions, which generally requires finding some normalizing constant Z_θ. This problem is infamously intractable. However, in using a Score-based objective, we are generally looking to optimize models of the form $p_\theta(x) = \dfrac{e^{-f_\theta(x)}}{Z_0}$, and then basing the objective on the Stein Score (importantly different than the Fisher Score) $\nabla_x \log p(x)$. In this case,

$$\nabla_x \log p(x) = -\nabla_x f(x) - \nabla_x \log Z_= - \nabla_x f(x) \tag{1}$$

where $\nabla_x \log Z = 0$ as it is not a function of x. Our objective then becomes $s_\theta = \nabla_x \log p_\theta(x) \approx \nabla_x \log p(x)$. A straightforward objective here would be the Fisher Divergence:

$$\frac{1}{2}\mathbb{E}_{p_{\text{data}}(x)}[||\nabla_x \log p_{\text{data}}(x) - s_\theta(x)||_2^2], \tag{2}$$

but this cannot be evaluated without access to $p(x)$ - the very object we are trying to recover. However, the technique of Score Matching allows us to compute this same norm via:

$$SM(\theta) = \mathbb{E}_{p_{\text{data}}(x)}[\frac{1}{2}||s_\theta(x)||_2^2 - \nabla_x \cdot s_\theta(x)], \tag{3}$$

Some pretty straightforward calculations will suffice to derive this equality (taken from Hyvärinen et al, 2005 [5]):

$$\begin{aligned} &||\nabla_x \log p_{\text{data}}(x) - s_\theta(x)||_2^2 \\ = (\nabla_x \cdot \log p_d(x))^2 - s_\theta(x) \cdot \nabla_x \log p_d(x)) + s_\theta(x)^2 \end{aligned}$$

where this first term is constant wrt θ. The quantity of interest here is the cross term, which we can massage out to the form:

$$\begin{aligned} \mathbb{E}_{p_{\text{data}}(x)}[-s_\theta(x) \cdot \nabla_x \log p_{\text{data}}(x)] = \\ -\int_{\mathcal{X}} s_\theta(x) \nabla_x \log p_{\text{data}}(x) p_{\text{data}}(\,dx) \\ = -\int_{\mathcal{X}} s_\theta(x) \frac{\nabla_x p_{\text{data}}(x)}{p_{\text{data}}(x)} p_{\text{data}}(x)\,dx \\ = -\int_{\mathcal{X}} s_\theta(x) \cdot \nabla_x p_{\text{data}}(x)\,dx \end{aligned}$$

And then using integration by parts:

$$\begin{aligned} &\mathbb{E}_{p_{\text{data}}(x)}[-s_\theta(x) \cdot \nabla_x \log p_{\text{data}}(x)] \\ = -s_\theta(x) \cdot p_{\text{data}}(x)\Big|_{-\infty}^{\infty} &+ \int_{\mathcal{X}} \nabla_x \cdot s_\theta(x) \nabla_x p_{\text{data}}(x)\, dx \\ &= \int_{\mathcal{X}} \nabla_x \cdot s_\theta(x) p_{\text{data}}(\, dx) \end{aligned}$$

Using a Theorem from Hyvärinen et al (2005) [5] that under mild regularity conditions, $p(x)s_\theta(x) \to 0$ as $||x||_2 \to \infty$. Bringing this together, we get

$$\mathbb{E}_{p_{\text{data}}(x)}[-s_\theta(x) \cdot \nabla_x \log p_{\text{data}}(x)] \propto \mathbb{E}_{p_{\text{data}}}[\nabla_x \cdot s_\theta(x) + \frac{1}{2}||s_\theta(x)||_2^2]$$

Luckily, Hyvärinen et al show that $L(\theta) = 0 \iff \theta = \theta^*$, so our straightforward NN optimization techniques should suffice.

Now looking into specific Diffusion Models for which such score matching may levergaged, we begin, unintuitvely, by looking into a likelihood based model. However, we will see that this example seamlesslly leads to the formulation of a Score-Based model of the same basic structure. Denoising Diffusion Probabalistics Models (DDPM), proposed by Ho et al in 2020 [3], consists of data sample $x_0 \sim p(x_0)$, with a forward and reverse Markov processes over a sequence $x_1, ..., x_T,$. The forward process contains a noise corrupting transition kernel $p(x_t|x_{t-1})$, and in the usual way, we may factorize the joint conditional distribution

$$p(x_1, ..., x_T|x_0) = \prod_{t=1}^{T} p(x_t|x_{t-1}) \tag{4}$$

Generally, this transition kernel is analytically derived to transition to a tractable prior distribution, most tractable of all being $\mathcal{N}(0, I)$, taking

$$p(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I) \tag{5}$$

Where $\beta_t \in (0, 1)$ is the called the Variance Scheduler, a hyperparameter. On the other hand, in the reverse process, wherein structure is restored to the corrupted data, a Gaussian transistion kernel is learned via some DNN. Here, we beging with $p(x_T) = \mathcal{N}(x_t; 0, I)$, which may be factored into

$$p_\theta(x_0, ..., x_T) = p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t), \qquad p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \tag{6}$$

When we sample from such a process, we want our learned Reverse Markov chain to match the actual time reversal as best we can, so we train by minimizing the Kullback-Leibler (KL) divergence between the two:

$$\text{KL}(p(x_0, ..., x_T)||p_\theta(x_0, ..., x_T)) \tag{7}$$

$$= \mathbb{E}\Big[-\log p(x_T) - \sum_{t\geq 1} \log \frac{p_\theta(x_{t-1}|x_t)}{p(x_t|x_{t-1})}\Big] = \mathbb{E}\Big[-\log \frac{p_\theta(x_0, ..., x_T)}{p(x_1, ..., x_T|x_0)}\Big] \tag{8}$$

$$= \mathbb{E}\Big[\underbrace{KL(p(x_T|x_0)||p(x_T))}_{\mathcal{L}_T} + \underbrace{\sum_{t>1} KL(p(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t))}_{\mathcal{L}_{t-1}} - \underbrace{\log p_\theta(x_0|x_1)}_{\mathcal{L}_0}\Big] \tag{9}$$

Where the step from (7) to (8) is called the reduced variational lower bound. See Appendix of Ho et al for more details. It turns out that during training, $\mathcal{L}_T$ and $\mathcal{L}_0$ can be ignored for optimization purposes. If we then let $\alpha_t := 1 - \beta_t, \bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$, we can re-parameterize $x_t(x_0, \epsilon) = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \hat{\alpha}_t}\epsilon$, $\epsilon \sim \mathcal{N}(0, 1)$ and $\mu_\theta(x_t, t)$ in terms of $\bar{\alpha}_t$ and ϵ_θ, DNN approximation of ϵ from x_t. With such a re-paramterization, Ho et al show our loss objective for $\mathcal{L}_{t-1}$ can be re-written

$$\arg\min_\theta \mathbb{E}_{t\sim U[1,T], x_0 \sim p(x_0), \epsilon \sim \mathcal{N}(0,I)} \left[\frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1 - \bar{\alpha}_t)} \| \epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \hat{\alpha}_t}\epsilon, t) \|^2 \right] \tag{10}$$

Upon arriving at this result, the authors note that this formulation "resembles denoising score matching over multiple noise scales indexed by t" [3]. Under such an interpretation, this DDPM process may be interpreted as an Annealed Langevin Dyanmics sampling procedure, a central algorithm for score based diffusion. This connection is driven home by Song et al 2019 [7]. Here, beginning with a score-matching weighted for a noise scheduler $\lambda(t)$, they derive

$$\mathbb{E}_{t\sim U[1,T], x_0 \sim p(x_0), x_t \sim p(x_t|x_0)} \left[\lambda(t) \sigma_t^2 \| \nabla_{x_t} p(x_t) - s_\theta(x_t, t) \|^2 \right] \tag{11}$$

$$= \mathbb{E}_{t\sim U[1,T], x_0 \sim p(x_0), x_t \sim p(x_t|x_0)} \left[\lambda(t) \left\| - \frac{x_t - x_0}{\sigma_t} - \sigma_t s_\theta(x_t, t)) \right\|^2 \right] + \text{constant} \tag{12}$$

$$= \mathbb{E}_{t\sim U[1,T], x_0 \sim p(x_0), \epsilon \sim \mathcal{N}(0,I)} \left[\lambda(t) \| \epsilon + \sigma_t s_\theta(x_t, t)) \|^2 \right] + \text{constant} \tag{13}$$

Where 7 is can be fully written in the form 11, letting $\lambda(t) = \dfrac{\beta_t^2}{2\sigma_t^2 \alpha_t (1 - \bar{\alpha}_t)}, \epsilon_\theta(x, t) = -\sigma_t s_\theta(x, t)$. With this explicit connection to score developed, such a model is called a Score-Based Generative Model (SGM). SGMs now produce samples from $s_\theta(x, T), ..., s_\theta(x, 0)$ using the Markov chains above with our reparamaterization of ϵ_θ, but now there is no model restrictions to enforce normalization.

2.3 SDE-Based Models

The SGM formulation above is hiding a simple insight which unlocks mathematical treasures. The above contained some finite number of steps $x_1, .., x_T$ in a continuous time reversible Markov Chain. However, if instead we want to instead look into a continuous sequence of random variables $\{x_t : t \in [0, T]\}$, with continuous noise schedulers β_t, σ_t, and allowing $p_t(x)$ denote the marginal probability density of x_t, we then enter the realm of Stochastic Differential Equations (SDEs). See **??** for a visualization of this process.

We treat our forward, noise corrupting process as the standard Ornstein–Uhlenbeck process:

$$dx_t^{\rightarrow} = \theta x_t \, dt + \sigma \, dW_t = \frac{1}{2}\beta(t) x dt + \sqrt{\beta(t)} dW_t \tag{14}$$

where W_t is standard Brownian motion. In Song et al (2020) [8], this is the corresponding SDE for the DDPM forward process. Note that this SDE can be analytically reversed. For an in depth proof of this fact, see Anderson, 1982 [1], however here, we will go through a heuristic derivation. Consider a discretization of the OU process above with $\delta << 1$. By Bayes' law, we have that

$$p(x_t \in dx | x_{t+\delta}) \propto p(x_t) \exp \left[- \frac{(x_{t+\delta} - (x_t + \mu_t \delta))^2}{2\sigma_t^2 \delta} \right] \tag{15}$$

With the exponential term accounting for discretized forward diffusion. Applying a 1st order Tayler expansion:

$$p(x_t) \approx p(x_{t+\delta}) \exp(\langle \nabla_{x_t} \log p(x_{t+\delta}), (x_t - x_{t+\delta}) \rangle) \tag{16}$$

and thus

$$p(x_t \in dx | x_{t+\delta}) \propto \exp\left[- \frac{(x_{t+\delta} - [(x_t + \mu_t \delta) + \sigma_t^2 \nabla_{x_t} \log p_t(x_{t+\delta})\delta])^2}{2\sigma_t^2 \delta} \right] \tag{17}$$

and so the transition of the reverse diffusion becomes

$$dx_t^{\leftarrow} = -\beta(t) \left[\frac{1}{2} + \nabla_{x_t} \log p(x_t) \right] + \sqrt{\beta(t)} dW_t \tag{18}$$

Notice how the score seamlessly emerges from this derivation. Thus, using Score Matching and a discretization of the SDEs above, we build a Diffusion Model as a continuous process analogue of the SGM developed above. See 1 for a visualization of this formalization. By formalizing these models as SDEs, we then get to benefit from the deep understanding accrued through decades of mathematical literature on SDEs. This allows us to get tighter bounds of performance than we could hope in the DDPM realm. For example, in Chen et al (2022) [2] we get some very powerful bounds on sampling accuracy, based on quality of score estimation. The centeral theorem of that paper contains three assumptions about an SGM:

1. The score function of the forward processes is L-Lipschitz.
2. The $(2 + \eta)$-th moment of p is finite, where $\eta > 0$ is arbitrarily small.
3. Data distribution p has finite KL-Divergence w.r.t. $\mathcal{N}(0, I)$.

Theorem 2.1 *Under assumptions (1-3), and if the the score estimator $s_\theta(x_t, t)$ has error in L^2 at most $\tilde{O}(\epsilon)$, then with appropriate step size, the SGM outputs a measure which is ϵ-close in total variation distance to p in $\tilde{O}(L^2 d/\epsilon)$ iterations.*

Much SDE theory is vital in proving this theorem within the paper cited. This concludes our brief survey over the mathematical developments of Diffusion Models.

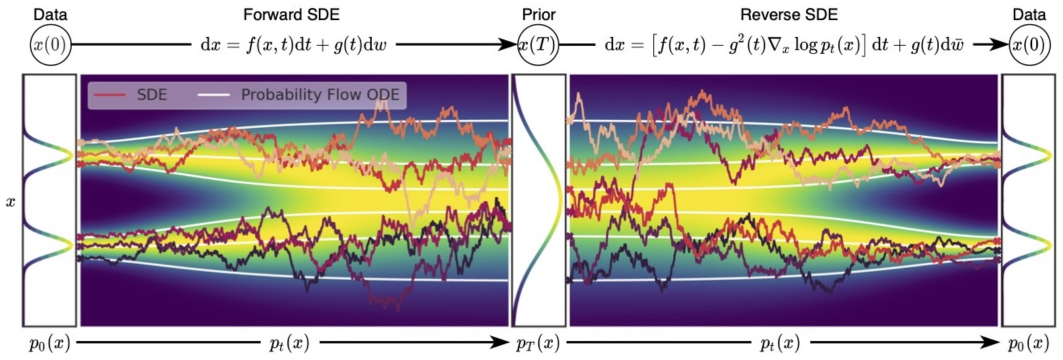
Data
Forward SDE
Prior
Reverse SDE
Data
$x(0)$
$\mathrm{d}x = f(x,t)\mathrm{d}t + g(t)\mathrm{d}w$
$x(T)$
$\mathrm{d}x = [f(x,t) - g^2(t)\nabla_x \log p_t(x)]\,\mathrm{d}t + g(t)\mathrm{d}\bar{w}$
$x(0)$
SDE
Probability Flow ODE
x
$p_0(x)$
$p_t(x)$
$p_T(x)$
$p_t(x)$
$p_0(x)$

Figure 1: Image from Song et al (2021)

3 Conditional generation

The vanilla generative model described above has limited practical utility, because it generates a random image, as opposed to following human's instructions. We describe two approaches to perform conditional generation, i.e. generating samples from the conditional distribution $p(x|c)$, where c is an image class, such as “cat" or “dog". Conditional generation allows the user to “steer" the model in test-time instead of having to train many different models, one trained on cat images, another one trained on dog images, etc.

3.1 Classifier Guidance

If one has access to a classifier $p(c|x)$ where c is a class and x is an image, then one can use Bayes' rule to compute

$$\nabla_x \log p(x|c) = \nabla_x \log p(c|x) + \nabla_x \log p(x) \tag{19}$$

since $p(c)$ is a constant wrt x. This suggests a way to turn an unconditional generative model into a conditional one by using a linear combination of $\nabla_x \log p(c|x)$ and $\nabla_x \log p(x)$ instead of $\nabla_x \log p(x)$ in the reverse process. In practice the following conditional score is used

$$\nabla_x \log p_\gamma(x|c) = (\gamma - 1)\nabla_x \log p(c|x) + \nabla_x \log p(x). \tag{20}$$

Note that $\gamma = -1$ corresponds to unconditional generation, whereas positive values of γ boost the guidance.

3.2 Classifier-Free Guidance

The approach in equation (20) suffers from two drawbacks: 1) automatic differentiation of the classifier is costly and noisy, 2) the accuracy of the guidance crucially depends on the accuracy of the classifier. Instead, one can evaluate $\nabla_x \log p(x|c)$ directly by modify the NN architecture to take the class c as input, along with image x and time t. In theory this is sufficient, at least for a finite set of classes c, but in practice a linear combination of the conditional score $\nabla_x \log p(x|c)$ and the unconditional score $\nabla_x \log p(x)$ is used. The unconditional score is obtained by masking the class in 20% of images in the train set. Thus, the classifier-free guidance is

$$\nabla_x \log p_\gamma(x|c) = (1 + \gamma) \cdot \nabla_x \log p(x|c) - \gamma \nabla_x \log p(x). \tag{21}$$

The negative sign may be surprising but (21) is derived from equation 20 via Bayes' rule. Intuitively, we guide the image away from the unconditional region and towards the conditional region. See [4] for more details on classifier-free guidance.

4 MNIST experiments

We performed several experiments on the MNIST dataset. We started with the Github repository mfkasim1/score-based-tutorial, which implements unconditional MNIST generation, and modified it to generate digits conditioned on the class $c \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. Our code is at Vilin97/score-based-tutorial.

4.1 Closest neighbors

To make sure that our generated digits are not exact copies of the training data, we computed the cosine similarity between the generated digits and the most similar images in the training data. The similarity is between 0.87 and 0.96, which indicates strong resemblance without memorization. See figure 2.

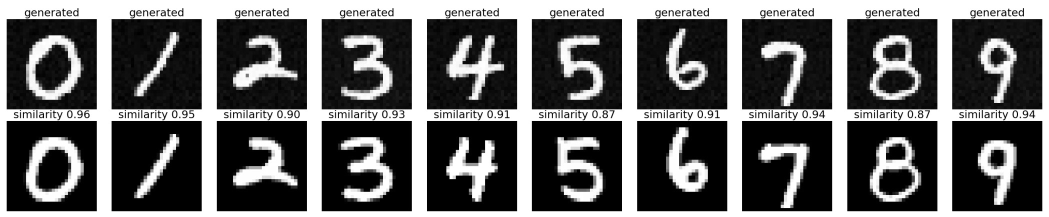

Figure 2: Generated digits (top row) and their closest neighbors in MNIST.

4.2 Time discretization

Additionally, we compared the effect of changing the number of time steps in the reverse diffusion process. See figure 3. We found that $n = 200$ strikes a good balance between quality and computational time, with $n = 20$ being really bad, $n = 50$ being recognizable but grainy, and $n = 1000$ only improving the quality by a little.

4.3 Guidance strength

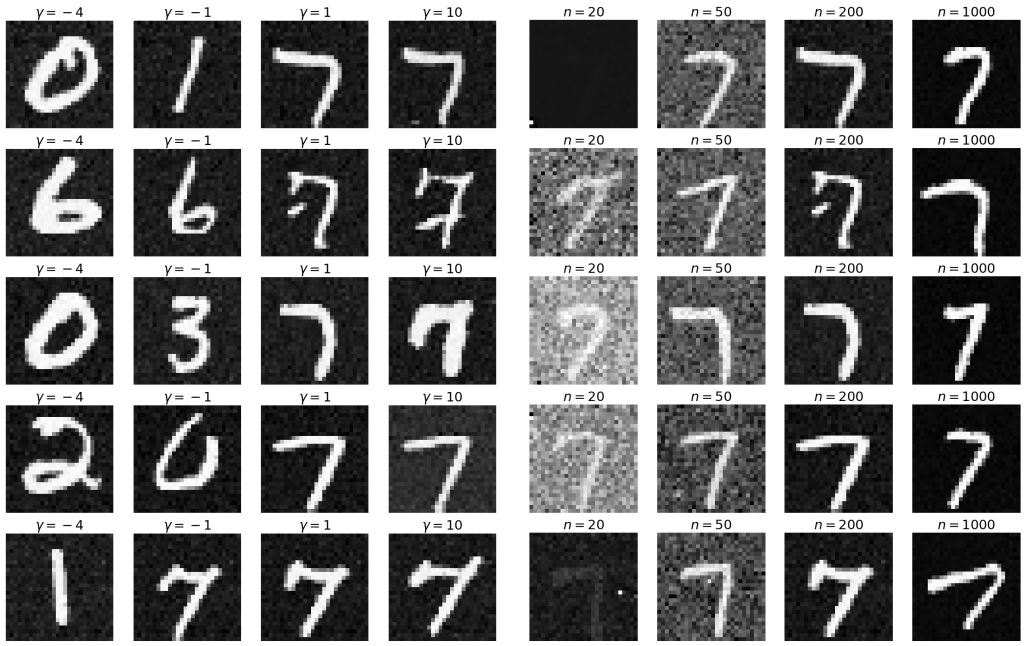

Figure 3: Effect of changing the guidance scale and time discretization.

We compared the effect of changing the guidance scale γ in equation (21).

If $\gamma = -1$, we get unconditional generation. When $\gamma > -1$, and especially $\gamma > 0$, the generation is conditional. When $\gamma < -1$, the generation is anti-conditional, which means that the conditioned class will never appear. In figure 3 the first column has no digit “7", because of the anti-conditioning, the second column contains a single “7" because of randomness, and the third and fourth columns contain all “7" because of the conditioning. Also, see figure 4 for a statistical analysis.

4.4 Digit Frequency

We compared the frequency of each digit in anti-conditional ($\gamma = -2$), unconditional ($\gamma = -1$) and conditional ($\gamma = -0.5, 0$) generations. We generated 1000 digits conditioned on “7", trained a 99%-accurate digit classifier and classified each generated digit. Since MNIST contains almost the same fraction of digits, the perfect **unconditional** ($\gamma = -1$) generative model will output each digit with roughly equal frequency. However, we do not observe this. See $\gamma = -1$ in figure 4. While MNIST is a toy dataset, this points to a big challenge in generative modeling – that the model can be biased even if the training data is unbiased.

When $\gamma = -2$, digit “7" is never generated, and digits that look similar such as “4" and “9" are rarely produced as well. When $\gamma = -0.5$, digit “7" is produced in 63% of all generations, and when $\gamma = 0$, it is produced in almost every generation.

Thus, the guidance strength allows the user to “guide" the generative model to produce or not produce a given class of images.

5 Implementation Details

We use an Encoder-Decoder U-Net with 16 layers, residual connections and LogSigmoid activations. We train with batch size 64 and learning rate 3e-4 for 2000 epochs. The training took one day on a single NVIDIA TITAN X Pascal GPU.

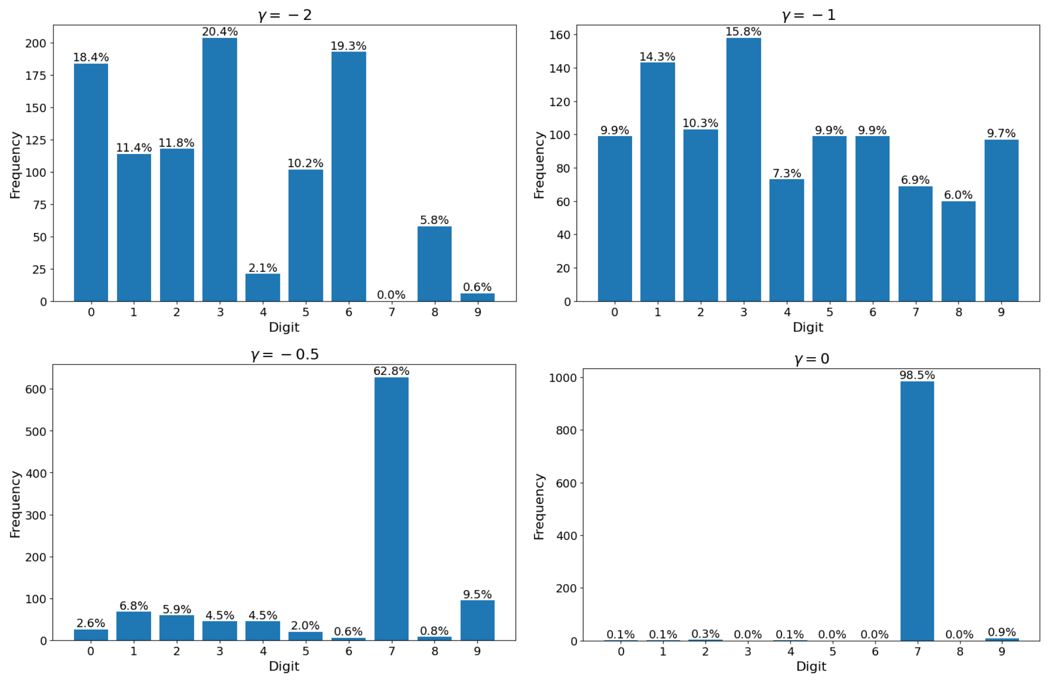

Figure 4: Comparison of digit frequencies conditioned on class “7" with different guidance strengths, anti-conditional ($\gamma = -2$), unconditional ($\gamma = -1$) and conditional ($\gamma = -0.5, 0$).

References

[1] Brian DO Anderson. “Reverse-time diffusion equation models”. In: *Stochastic Processes and their Applications* 12.3 (1982), pp. 313–326.

[2] Sitan Chen, Sinho Chewi, Jerry Li, Yuanzhi Li, Adil Salim, and Anru R Zhang. “Sampling is as easy as learning the score: theory for diffusion models with minimal data assumptions”. In: *arXiv preprint arXiv:2209.11215* (2022).

[3] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising diffusion probabilistic models”. In: *Advances in neural information processing systems* 33 (2020), pp. 6840–6851.

[4] Jonathan Ho and Tim Salimans. “Classifier-free diffusion guidance”. In: *arXiv preprint arXiv:2207.12598* (2022).

[5] Aapo Hyvärinen and Peter Dayan. “Estimation of non-normalized statistical models by score matching.” In: *Journal of Machine Learning Research* 6.4 (2005).

[6] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. “Deep unsupervised learning using nonequilibrium thermodynamics”. In: *International conference on machine learning*. PMLR. 2015, pp. 2256–2265.

[7] Yang Song and Stefano Ermon. “Generative modeling by estimating gradients of the data distribution”. In: *Advances in neural information processing systems* 32 (2019).

[8] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. “Score-based generative modeling through stochastic differential equations”. In: *arXiv preprint arXiv:2011.13456* (2020).