Deep Equilibrium Approaches to Diffusion Models

Anonymous Author(s)
Affiliation
Address
email

Abstract

Diffusion-based generative models have shown to be extremely effective in generating high-quality images, with generated samples often surpassing the quality of those produced by other models under several metrics. One distinguishing feature of these models, however, is that they typically require long sampling chains in order to produce high-fidelity images. This presents a challenge not only from the lenses of sampling time, but also from the inherent difficulty in backpropagating through these chains in order to accomplish tasks such as model inversion, *i.e.,* approximately finding latent states that generate known images. In this paper, we look at diffusion models through a different perspective, that of a (deep) equilibrium (DEQ) fixed point model. Specifically, we extend the recent denoising diffusion implicit model (DDIM) [59], and model the entire sampling chain as a joint, multivariate fixed point system. This setup provides an elegant unification of diffusion and equilibrium models, and shows benefits in 1) single image sampling, as it replaces the fully-serial typical sampling process with a parallel one; and 2) model inversion, where we can leverage fast gradients in the DEQ setting to much more quickly find the noise that generates a given image. The approach is also orthogonal and thus complementary to other methods used to reduce the sampling time, or improve model inversion. We demonstrate our method's strong performance across several datasets, including CIFAR10, CelebA, and LSUN Bedroom and Churches.

1 Introduction

Diffusion models have emerged as a promising class of generative models that can generate high quality images [60, 59, 50], outperforming GANs on perceptual quality metrics [16], and likelihood-based models on density estimation [37]. One of the limitations of these models, however, is the fact that they require a long diffusion chain (many repeated applications of an denoising process), in order to generate high-fidelity samples. Several recent papers have focused on tackling this limitation, *e.g.,* by shortening the length of diffusion process through an alternative parameterization [59, 39], or through progressive distillation of a sampler with large diffusion chain into a smaller one [47, 56]. However, all of these methods still rely on a fundamentally sequential sampling process, imposing challenges on accelerating the sampling and for other applications like differentiating through the entire generation process.

In this paper, we propose an alternative approach that also begins to address such challenges from a different perspective. Specifically, we propose to model the generative process of a specific class of diffusion model, the denoising diffusion implicit model (DDIM) [59], as a deep equilibrium (DEQ) model. Deep equilibrium (DEQ) models are networks that aim to find the fixed point of the underlying system in the forward pass and differentiate implicitly through this fixed point in the backward pass.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

To apply DEQ to diffusion models, we first formulate this process as an equilibrium system consisting of all T joint sampling steps, and then *simultaneously* solve the fixed point of all the T steps.

This approach has several benefits: **First**, the DEQ sampling process can be solved in parallel over multiple GPUs by scattering the workload. This is particularly beneficial in the case of single image (*i.e.,* batch-size-one) generation, where the serial sampling nature of diffusion models has inevitably made them unable to maximize GPU computation. **Second**, solving for the joint equilibria simultaneously leads to faster overall convergence as we have better estimates of the intermediate states in fewer steps. Specifically, the formulation naturally lends itself to an augmented diffusion chain that each state is generated according to *all* others. **Third**, the DEQ formulation allows us to leverage a faster differentiation through the chain. This enables us to much more effectively solve problems that require us to differentiate through the generative process, useful for tasks such as *model inversion* that seeks to find the noise that leads to a particular instance of an image.

We demonstrate the advantages of this formulation on two applications: single image generation and model inversion. Both single image generation and model inversion are widely applied in real world image manipulation tasks like image editing and restoration [67, 54, 30, 1, 43]. On CIFAR-10 [41] and CelebA [45], DEQ achieves up to $2\times$ speedup over the sequential sampling processes of DDIM [59], while maintaining a comparable perceptual quality of images. In the model inversion, the loss converges much faster when trained with DEQs than with sequential sampling. Moreover, optimizing the sequential sampling process can be computationally expensive. Leveraging modern autograd packages is infeasible as they require storing the entire computational graph for all T states. Some recent works like Nie et al. [51] achieve this through use of SDE solvers. In contrast, with DEQs we can use the implicit differentiation of $\mathcal{O}(1)$ memory complexity. Empirically, the initial hidden state recovered by DEQ more accurately regenerates the original image while capturing its finer details.

To summarize, our main contributions are as follows:

- We formulate the generative process of an augmented type of DDIM as a deep equilibrium model that allows the use of black-box solvers to efficiently compute the fixed point and generate images.
- The DEQ formulation parallelizes the sampling process of DDIM, and as a result, it can be run on multiple GPUs instead of a single GPU. This alternate sampling process converges faster compared to the original process.
- We demonstrate the advantages of our formulation on single image generation and model inversion. We find that optimizing the sampling process via DEQs consistently outperforms naive sequential sampling.

2 Preliminaries

Diffusion Models Denoising diffusion probabilistic models (DDPM) [58, 29] are generative models that can convert the data distribution to a simple distribution, (*e.g.,* a standard Gaussian, $\mathcal{N}(\mathbf{0}, \mathbf{I})$), through a diffusion process. Specifically, given samples from a target distribution $\mathbf{x}_0 \sim q(\mathbf{x}_0)$, the diffusion process is a Markov chain that adds Gaussian noise to the data to generate latent states $\mathbf{x}_1, ..., \mathbf{x}_T$ in the same sample space as $\mathbf{x}_0$. The inference distribution of diffusion process is given by:

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \tag{1}$$

To learn the parameters θ that characterize a distribution $p_\theta(\mathbf{x}_0) = \int p_\theta(\mathbf{x}_{0:T}) d\mathbf{x}_{1:T}$ as an approximation of $q(\mathbf{x}_0)$, a surrogate variational lower bound [58] was proposed to train this model:

$$L = \mathbb{E}_q[-\log p_\theta(\mathbf{x}_0|\mathbf{x}_1) + \sum_t D_{KL}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) + D_{KL}(q(\mathbf{x}_T|\mathbf{x}_0)||p(\mathbf{x}_T)] \tag{2}$$

After training, samples can be generated by a reverse Markov chain, *i.e.,* first sampling $\mathbf{x}_T \sim p(\mathbf{x}_T)$, and then repeatedly sampling $\mathbf{x}_{t-1}$ till we reach $\mathbf{x}_0$.

As noted in [58, 59], the length T of a diffusion process is usually large (e.g. $T = 1000$ [29]) as it contributes to a better approximation of Gaussian conditional distributions in the generative process. However, because of the large value of T, sampling from diffusion models can be visibly slower compared to other deep generative models like GANs [25].

One feasible acceleration is to rewrite the forward process into a non-Markovian one that leads to a "shorter" and deterministic generative process, *i.e.,* denoising diffusion implicit model [59] (DDIM). DDIM can be trained similarly to DDPM, using the variational lower bound shown in Eq. (2). Essentially, DDIM constructs a nearly non-stochastic scheme that can quickly sample from the learned data distribution without introducing additional noises. Specifically, the scheme to generate a sample $\mathbf{x}_{t-1}$ given $\mathbf{x}_t$ is:

$$\mathbf{x}_{t-1} = \sqrt{\alpha_{t-1}}\left(\frac{\mathbf{x}_t - \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_\theta^{(t)}(\mathbf{x}_t)}{\sqrt{\alpha_t}}\right) + \sqrt{1-\alpha_{t-1}-\sigma_t^2}\cdot\boldsymbol{\epsilon}_\theta^{(t)}(\mathbf{x}_t) + \sigma_t\boldsymbol{\epsilon}_t \tag{3}$$

where $\alpha_1, ..., \alpha_T \in (0, 1]$, $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and $\boldsymbol{\epsilon}_\theta^{(t)}(\mathbf{x}_t)$ is an estimator trained to predict the noise given a noisy state $\mathbf{x}_t$. Different values of σ_t define different generative processes. For a variance schedule $\beta_1 \dots \beta_T$, we use the notation $\alpha_t = \prod_{s=1}^{t}(1-\beta_s)$. When $\sigma_t = \sqrt{(1-\alpha_{t-1})/(1-\alpha_t)}\sqrt{1-\alpha_t/\alpha_{t-1}}$ for all t, the generative process represents a DDPM. Setting $\sigma_t = 0$ for all t gives rise to a DDIM, which results in a deterministic generating process except the initial sampling $\mathbf{x}_T \sim p(\mathbf{x}_T)$.

Deep Equilibrium Models Deep equilibrium models are a recently-proposed class of deep networks that, in their forward pass, seek to find a fixed point of a single layer applied repeatedly to a hidden state. Specifically, consider a deep feedforward model with L layers:

$$\mathbf{z}^{[i+1]} = f_\theta^{[i]}\left(\mathbf{z}^{[i]}; \mathbf{x}\right) \quad \text{for } i = 0, ..., L-1 \tag{4}$$

where $\mathbf{x}$ is the input variable, $\mathbf{z}^{[i]}$ is the hidden state of ith layer, and $f_\theta^{[i]}$ is a layer that defines the feature transformation. Assuming the above model is weight-tied, *i.e.,* $f_\theta^{[i]} = f_\theta, \forall i$, then in the limit of infinite depth, the output $\mathbf{z}^{[i]}$ of this network converges to a fixed point $\mathbf{z}^*$.

$$\lim_{i\to\infty} f_\theta\left(\mathbf{z}^{[i]}; \mathbf{x}\right) = \mathbf{z}^* \tag{5}$$

Inspired from the neural convergence phenomenon, Deep equilibrium (DEQ) models [5] are proposed to directly compute this fixed point $\mathbf{z}^*$ as the output, *i.e.,*

$$f_\theta(\mathbf{z}^*; \mathbf{x}) = \mathbf{z}^* \tag{6}$$

The equilibrium state $\mathbf{z}^*$ can be solved by black-box solvers like Broyden's method [11], or Anderson acceleration [4]. To train this fixed-point system, Bai et al. [5] leverage implicit differentiation to directly backpropagate through the equilibrium state $\mathbf{z}^*$ using $\mathcal{O}(1)$ memory complexity. DEQ is known as a principled framework for characterizing convergence and energy minimization in deep learning. We leave a detailed discussion in Sec. 6.

3 A Deep Equilibrium Approach to DDIMs

In this section, we present the main modeling contribution of the paper, a formulation of diffusion processes under the DEQ framework. Although diffusion models may seem to be a natural fit for DEQ modeling (after all, we typically *do not* care about intermediate states in the denoising chain, but only the final clean image), there are several reasons why setting up the diffusion chain "naively" as a DEQ (*i.e.,* making f_θ be a single sampling step) does not ultimately lead to a functional algorithm. Most fundamentally, the diffusion process is not time-invariant (*i.e.,* not "weight-tied" in the DEQ sense), and the final generated image is practically-speaking independent of the noise used to generate it (*i.e.,* not truly based upon "input injection" either).

Thus, at a high level, our approach to building a DEQ version of the DDIM involves representing *all* the states $\mathbf{x}_{0:T}$ *simultaneously* within the DEQ state. The advantage of this approach is that 1) we can exactly capture the typical diffusion inference chain; and 2) we can create a *more expressive* reverse process where the state $\mathbf{x}_t$ is updated based upon *all* later states $\mathbf{x}_{t+1:T}$, improving the inference process. A notable *downside* of this representation is that 1) we need to store all states simultaneously (but *only* the images, and not the intermediate network states, so that this is not overly constraining); 2) we can execute all steps of the inference chain *in parallel* rather than solely in sequence as is typically required in diffusion models; and 3) we can use common DEQ acceleration methods, such as the Anderson solver [4] to find the fixed point, which makes the sampling process converge faster.

A DEQ formulation of DDIMs. The generative process of DDIM is given by:

$$\mathbf{x}_{t-1} = \sqrt{\alpha_{t-1}}\left(\frac{\mathbf{x}_t - \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_\theta^{(t)}(\mathbf{x}_t)}{\sqrt{\alpha_t}}\right) + \sqrt{1-\alpha_{t-1}} \cdot \boldsymbol{\epsilon}_\theta^{(t)}(\mathbf{x}_t), \quad t = [1, \dots, T] \tag{7}$$

This process also lets us generate a sample using a subset of latent states $\{\mathbf{x}_{\tau_1}, \dots, \mathbf{x}_{\tau_S}\}$, where $\{\tau_1, \dots, \tau_S\} \subseteq T$. While this helps in accelerating the overall generative process, there is a tradeoff between sampling quality and computational efficiency. As noted in Song et al. [59], larger T lead to lower FID scores of the generated images but need more compute time; smaller T are faster to sample from, but the resulting images have worse FID scores.

Reformulating this sampling process as a DEQ addresses multiple concerns raised above. We can define a DEQ, with a sequence of latent states $\mathbf{x}_{1:T}$ as its internal state, that *simultaneously* solves for the equilibrium points at all the timesteps. The global convergence of this process is upper bounded by T steps, by definition. To derive the DEQ formulation of the generative process, first we rearrange the terms in Eq. (7):

$$\mathbf{x}_{t-1} = \sqrt{\frac{\alpha_{t-1}}{\alpha_t}}\mathbf{x}_t + \left(\sqrt{1-\alpha_{t-1}} - \sqrt{\frac{\alpha_{t-1}(1-\alpha_t)}{\alpha_t}}\right)\epsilon_\theta^{(t)}(\mathbf{x}_t) \tag{8}$$

Let $c_1^{(t)} = \sqrt{1-\alpha_{t-1}} - \sqrt{\frac{\alpha_{t-1}(1-\alpha_t)}{\alpha_t}}$. Then we can write

$$\mathbf{x}_{t-1} = \sqrt{\frac{\alpha_{t-1}}{\alpha_t}}\mathbf{x}_t + c_1^{(t)}\epsilon_\theta^{(t)}(\mathbf{x}_t) \tag{9}$$

By induction, we can rewrite the above equation as:

$$\mathbf{x}_{T-k} = \sqrt{\frac{\alpha_{T-k}}{\alpha_T}}\mathbf{x}_T + \sum_{t=T-k}^{T-1}\sqrt{\frac{\alpha_{T-k}}{\alpha_t}}c_1^{(t+1)}\epsilon_\theta^{(t+1)}(\mathbf{x}_{t+1}), \quad k \in [0, .., T] \tag{10}$$

This defines a "fully-upper-triangular" inference process, where the update of $\mathbf{x}_t$ depends on the noise prediction network ϵ_θ applied to *all* subsequent states $\mathbf{x}_{t+1:T}$; in contrast to the traditional diffusion process, which updates $\mathbf{x}_t$ based only on $\mathbf{x}_{t+1}$. Specifically, let $h(\cdot)$ represent the function that performs the operations in the equations (10) for a latent $\mathbf{x}_t$ at timestep t, and let $\tilde{h}(\cdot)$ represent the function that performs the same set of operations across all the timesteps simultaneously. We can write the above set of equations as a fixed point system:

$$\begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{x}_{T-2} \\ \vdots \\ \mathbf{x}_0 \end{bmatrix} = \begin{bmatrix} h(\mathbf{x}_T) \\ h(\mathbf{x}_{T-1}, \mathbf{x}_T) \\ \vdots \\ h(\mathbf{x}_1, ..., \mathbf{x}_T) \end{bmatrix}$$

or,

$$\mathbf{x}_{0:T-1} = \tilde{h}(\mathbf{x}_{0:T-1}; \mathbf{x}_T) \tag{11}$$

The above system of equations represent a DEQ with $\mathbf{x}_T$ as input injection. We can simultaneously solve for the roots of this system of equations through black-box solvers like Anderson acceleration [4]. Let $g(\mathbf{x}_{0:T-1}; \mathbf{x}_T) = \tilde{h}(\mathbf{x}_{0:T-1}; \mathbf{x}_T) - \mathbf{x}_{0:T-1}$, then we have

$$\mathbf{x}^*_{0:T} = \text{Solver}(g(\mathbf{x}_{0:T-1}); \mathbf{x}_T) \tag{12}$$

This DEQ formulation has multiple benefits. Solving for all the equilibria simultaneously leads to a better estimation of the intermediate latent states $\mathbf{x}_t$ in a fewer number of steps (*i.e.,* $\leq t$ steps for $\mathbf{x}_t$). This leads to faster convergence of the sampling process as the final sample $\mathbf{x}_0$, which is dependent on the latent states of all the previous time steps, has a better estimate of these intermediate latent states. Note that by the same reasoning, the intermediate latent states $\mathbf{x}_t$ converge faster too. Thus, we can get images with perceptual quality comparable to DDIM in a significantly fewer number of steps. Of course, we also note that the computational requirements of each individual step has significantly increased, but this is at least largely offset by the fact that the steps can be executed as mini-batched in parallel over each state. Empirically, in fact, we often notice significant *speedup* using this approach on tasks like single image generation.

4 Efficient Inversion of DDIM

One of the primary strengths of DEQs is their constant memory consumption, for both forward pass and backward pass, regardless of their 'effective depth'. This leads to an interesting application of DEQs in inverting DDIMs that fully leverages this advantage along with the other benefits discussed in the previous section.

Algorithm 1 A naive algorithm to invert DDIM

Input: A target image $\mathbf{x}_0 \sim \mathcal{D}$, $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$ a trained denoising diffusion model, N the total number of epochs
▷ f denotes the sampling process in Eq (7)
Initialize $\hat{\mathbf{x}}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for epochs from 1 to N **do**
  for $t = T, ..., 1$ **do**
    Sample $\hat{\mathbf{x}}_{t-1} = f(\hat{\mathbf{x}}_t; \epsilon_\theta(\hat{\mathbf{x}}_t, t))$
  end for

  Take a gradient descent step on

$$\nabla_{\hat{\mathbf{x}}_T} \|\hat{\mathbf{x}}_0 - \mathbf{x}_0\|_F^2$$

end for
Output: $\hat{\mathbf{x}}_0$

Algorithm 2 Inverting DDIM with DEQ

Input: A target image $\mathbf{x}_0 \sim \mathcal{D}$, $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$ a trained denoising diffusion model, N the total number of epochs
▷ g is the function in Eq. (12)
Initialize $\hat{\mathbf{x}}_{0:T} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for epochs from 1 to N **do**
  ▷ Disable gradient computation
  $\mathbf{x}^*_{0:T} = \text{RootSolver}(g(\mathbf{x}_{0:T-1}); \mathbf{x}_T)$
  ▷ Enable gradient computation
  Compute Loss $\mathcal{L}(\mathbf{x}_0, \mathbf{x}^*_0)$
  Use the 1-step grad to compute $\dfrac{\partial \mathcal{L}}{\partial \mathbf{x}_T}$
  Take a gradient descent step using above
end for
Output: $\mathbf{x}^*_0$

4.1 Problem Setup

Given an arbitrary image $\mathbf{x}_0 \sim \mathcal{D}$, and a denoising diffusion model $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$ trained on a dataset $\mathcal{D}$, model inversion seeks to determine the latent $\hat{\mathbf{x}}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ that can generate an image $\hat{\mathbf{x}}_0$ identical to the original image $\mathbf{x}_0$ through the generative process for DDIM described in Eq. (7). For an input image $\mathbf{x}_0$, and a generated image $\hat{\mathbf{x}}_0$, this task needs to minimize the squared-Frobenius distance between these images:

$$\mathcal{L}(\mathbf{x}_0, \hat{\mathbf{x}}_0) = \|\mathbf{x}_0 - \hat{\mathbf{x}}_0\|_F^2 \tag{13}$$

4.2 Inverting DDIM: A Naive Way

A relatively straightforward way to invert DDIM is to randomly sample $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and update it via gradient descent by first estimating $\mathbf{x}_0$ using the generative process in Eq. (7) and backpropagating through this process after computing the loss objective in (13). The overall process has been summarized in Algorithm 1. This process has a large computational overhead. Every training epoch

requires a sequential sampling for all T timesteps. Optimizing through this generative process would require the creation of a large computational graph for storing relevant intermediate variables necessary for the backward pass. Sequential sampling further slows down the entire process.

4.3 Efficient Inversion of DDIM with DEQs

Alternatively, we can use the DEQ formulation to develop a much more efficient inversion method. We provide a high-level overview of this approach in Algorithm 2. We can apply implicit function theorem (IFT) to the fixed point, *i.e.,* (12) to compute gradients of the loss $\mathcal{L}(\mathbf{x}_0, \mathbf{x}_0^*)$ in (13) *w.r.t.* $(\cdot)$:

$$\frac{\partial \mathcal{L}}{\partial(\cdot)} = -\frac{\partial \mathcal{L}}{\partial \mathbf{x}_{0:T}^*}\left(J_{g_\theta}^{-1}\Big|_{\mathbf{x}_{0:T}^*}\right)\frac{\partial \tilde{h}(\mathbf{x}_{0:T-1}^*; \mathbf{x}_T)}{\partial(\cdot)} \tag{14}$$

where $(\cdot)$ could be any of the latent states $\mathbf{x}_1, ..., \mathbf{x}_T$, and $J_{g_\theta}^{-1}\Big|_{\mathbf{x}_{0:T}^*}$ is the inverse Jacobian of $g(\mathbf{x}_{0:T-1}; \mathbf{x}_T)$ evaluated at $x_{0:T}^*$. Refer to [5] for a detailed proof. Computing the inverse of Jacobian matrix can become computationally intractable, especially when the latent states $\mathbf{x}_t$ are high dimensional. Further, prior works [5, 7, 24] have reported growing instability of DEQs during training due to the ill-conditioning of Jacobian. Recent works [23, 22, 24, 8] suggest that we do not need an exact gradient to train DEQs. We can instead use an approximation to Eq. (14), *i.e.,*

$$\frac{\partial \mathcal{L}}{\partial(\cdot)} = -\frac{\partial \mathcal{L}}{\partial \mathbf{x}_{0:T}^*}\mathbf{M}\frac{\partial \tilde{h}(\mathbf{x}_{0:T-1}^*; \mathbf{x}_T)}{\partial(\cdot)} \tag{15}$$

where $\mathbf{M}$ is an approximation of $J_{g_\theta}^{-1}\Big|_{\mathbf{x}_{0:T}^*}$. For example, [23, 22, 24] show that setting $\mathbf{M} = \mathbf{I}$, *i.e.,* 1-step gradient, works well. In this work, we follow Geng et al. [24] to further add a damping factor to the 1-step gradient. The forward pass is given by:

$$\mathbf{x}_{0:T}^* = \text{Solver}(g(\mathbf{x}_{0:T-1}); \mathbf{x}_T) \tag{16}$$

$$\mathbf{x}_{0:T}^* = \tau \cdot \tilde{h}(\mathbf{x}_{0:T-1}^*; \mathbf{x}_T^*) + (1-\tau) \cdot \mathbf{x}_{0:T}^* \tag{17}$$

The gradients for the backward pass can be computed through standard autograd packages. We provide the PyTorch-style pseudocode of our approach in the Appendix. Using inexact gradients for the backward pass has several benefits: 1) It remarkably improves the training stability of DEQs; 2) Our backward pass consists of a single step and is ultra-cheap to compute. It reduces the total training time by a significant amount.

5 Experiments

We consider four datasets that have images of different resolutions for our experiments: CIFAR10 (32×32) [41], CelebA (64×64) [45], LSUN Bedroom (256×256) and LSUN Outdoor Church (256×256) [66]. For all the experiments, we use Anderson acceleration as the default fixed point solver. We use the pretrained denoising diffusion models from Ho et al. [29] for CIFAR10, LSUN Bedroom, and LSUN Outdoor Church, and from Song et al. [59] for CelebA. While training DEQs for model inversion, we use the 1-step gradient Eq. (17) to compute the backward pass. The damping factor τ for 1-step gradient is set to 0.1. All the experiments have been performed on NVIDIA RTX A6000 GPUs. We provide additional experiment details in the Appendix. Code will be released along with the paper.

5.1 Convergence of DDIM with DEQs

We verify that DEQ converges to a fixed point by plotting the values of $\|\tilde{h}(\mathbf{x}_{0:T}) - \mathbf{x}_{0:T}\|_2$ over Anderson solver steps. As seen in Fig. 10, DEQ converges to a fixed point for generative processes of different lengths. It is easier to reach simultaneous equilibria on smaller sequence lengths than larger sequence lengths. However, this does not affect the quality of images generated. We visualize the latent states of DEQ in Fig. 2. Our experiments demonstrate that DEQ is able to generate high-quality images in as few as 15 Anderson solver steps on diffusion chains that were trained on a

much larger number of steps T. One might note that DEQs converge to a limit cycle for diffusion processes with larger sequence lengths. This is not a limitation as we only want the latent states at the last few timesteps to converge well, which happens in practice as demonstrated in Fig. 2. Further, these residuals can be driven down by using more powerful solvers like quasi-Newton methods, *e.g.,* Broyden's method.

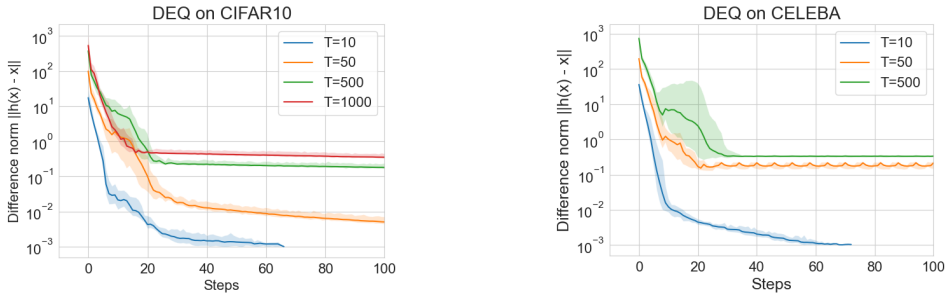

Figure 1: DEQ formulation of DDIM finds an equilibrium point. We plot the absolute fixed-point convergence $\|\tilde{h}(\mathbf{x}_{0:T}) - \mathbf{x}_{0:T}\|_2$ during a forward pass of DEQ for CIFAR-10 (left) and CelebA (right) for different number of steps T. The shaded region indicates the maximum and minimum value encountered during any of the 25 runs.

Figure 2: Visualization of intermediate latents $\mathbf{x}_t$ after 15 forward steps with Anderson solver for CIFAR-10 (first row, $T = 500$), CelebA (second row, $T = 500$), LSUN Bedroom (third row, $T = 50$, and LSUN Outdoor Church (fourth row, $T = 50$). For $T = 500$, we visualize every 50th latent, and for $T = 50$, we visualize every 5th latent. In addition, we also visualize $\mathbf{x}_{0:4}$ in the last 5 columns.

Dataset	T	DDPM		DDIM		DEQ	
		FID	Time	FID	Time	FID	Time
CIFAR10	1000	**3.17**	24.45s	4.07	20.16s	3.79	**2.91s**
CelebA	500	5.32	14.95s	3.66	10.31s	**2.92**	**5.12s**
LSUN Bedroom	25	184.05	1.72s	8.76	**1.19s**	**8.73**	3.82s
LSUN Church	25	122.18	1.77s	**13.24**	**1.68s**	13.62	3.99s

Table 1: FID scores and time for single image generation for DDPM, DDIM and DEQ.

5.2 Sample quality of images generated with DEQ

We verify that DEQs can generate images of comparable quality to DDIM by reporting Fréchet Inception Distance (FID) [28] in Table 1. For the forward pass of DEQ, we run Anderson solver for a maximum of 15 steps for each image. We report FID scores on 50,000 images, and average time to generate an image (including GPU time) on 500 images. We note significant gains in wall-clock time on single-shot image generation with DEQs on images with smaller resolutions. Specifically, DEQs can generate images almost 2× faster than the sequential sampling of DDIM on CIFAR-10 (32×32) and CelebA (64×64). We note that these gains vanish on sequences of shorter lengths,

and on images with larger resolutions as seen in case of LSUN Bedrooms, and Outdoor Churches (256×256). We also find that on full-batch inference with larger batches, sequential sampling might outperform DEQs, as DEQs would have larger memory requirements in this case, *i.e.,* processing smaller batches of size B might be faster than processing larger batches of size BT.

5.3 Model Inversion of DDIM with DEQs

We report the minimum values of squared Frobenius norm between the recovered and target images averaged from 100 different runs in Table 2. DEQ outperforms the baseline method on all the datasets by a significant margin. We also plot the training loss curves of DEQ and the baseline in Fig. 3. We observe that DEQ converges faster and has much lower loss values than the baseline method induced by DDIM. We also visualize the images generated with the recovered latent states in Fig. 4. It is worth noting that images generated with DEQs capture more vivid details of the original images, like textures of foliage, crevices, and other finer details than the baseline.

Dataset	T	Baseline		DEQ	
		Min loss ↓	Avg Time (mins) ↓	Min loss ↓	Avg Time (mins) ↓
CIFAR10	100	15.74 ± 8.7	49.07 ± 1.76	$\mathbf{0.76 \pm 0.3}5$	12.99 ± 0.97
CIFAR10	10	2.59 ± 3.67	14.36 ± 0.26	$\mathbf{0.68 \pm 0.32}$	2.54 ± 0.41
CelebA	20	14.13 ± 5.04	30.09 ± 0.57	$\mathbf{1.03 \pm 0.37}$	28.09 ± 1.76
Bedroom	10	1114.49 ± 795.86	26.41 ± 0.17	$\mathbf{36.37 \pm 22.86}$	33.7 ± 1.05
Church	10	1674.68 ± 1432.54	29.7 ± 0.75	$\mathbf{47.94 \pm 24.78}$	33.54 ± 3.02

Table 2: Comparison of minimum loss and average time required to generate an image. All the numbers have been reported on 100 images. See Appendix for detailed training settings.

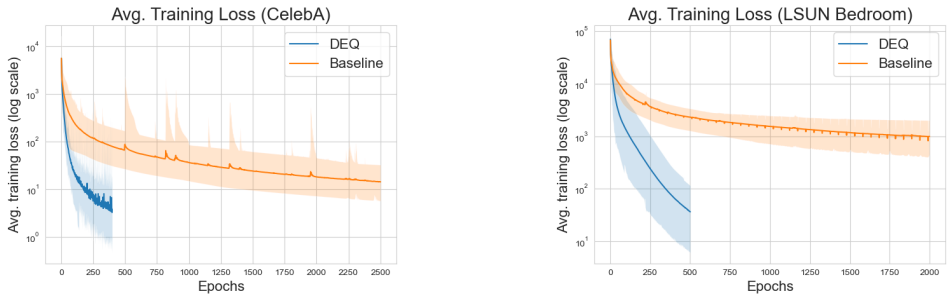

Figure 3: Training loss for CelebA and LSUN Bedroom over epochs. DEQ converges in fewer epochs, and achieves lower values of loss compared to the baseline. The shaded region indicates the maximum and minimum value of loss encountered during any of the 100 runs.

6 Related Work

Implicit Deep Learning Implicit deep learning is an emerging field that introduces structured methods to construct modern neural networks. Different from prior explicit counterparts defined by hierarchy or layer stacking, implicit models take advantage of dynamical systems [38, 20, 2], *e.g.,* optimization [3, 63, 17, 23, 18], differential equation [13, 19, 61, 26], or fixed-point system [5, 6, 27]. For instance, Neural ODE [13] describes a continuous time-dependent system, while Deep Equilibrium (DEQ) model [5], which is actually path-independent, is a new type of implicit models that outputs the equilibrium states of the underlying system, *e.g.,* $\mathbf{z}^*$ from $\mathbf{z}^* = f_\theta(\mathbf{z}^*, \mathbf{x})$ given the input $\mathbf{x}$. This fixed-point system can be solved by black-box solvers [4, 11], and further accelerated by the neural solver [9] in the inference. An active research topic is the stability [7, 24, 8] of such a system as it will gradually deteriorate during training, albeit having strong performance [13, 5, 7]. DEQ has achieved SOTA results on a wide-range of tasks like language modeling [5], semantic segmentation [6], graph modeling [27, 44], object detection [64], optical flow estimation [8], and generative models like normalizing flow [46], with theoretical guarantees [33, 65].

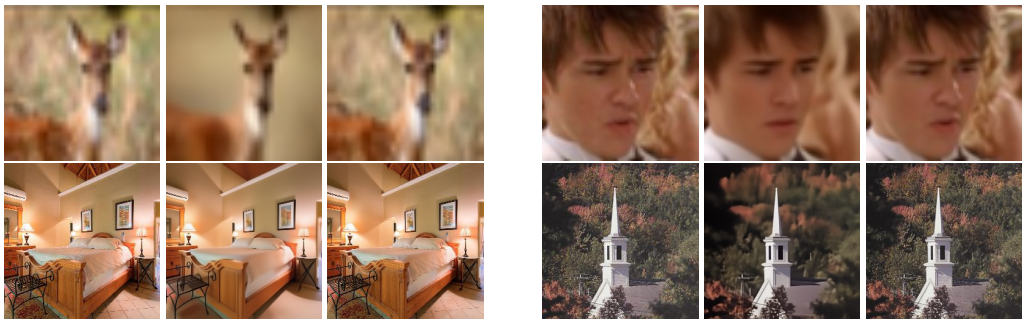

Figure 4: Model inversion on CIFAR10, CelebA, LSUN Bedrooms and Churches, respectively. Each triplet has the original image (left), DDIM's inversion (middle), and DEQ's inversion (right).

Diffusion Models Diffusion models [58, 29, 59], or score-based generative models [60, 62], are newly developed generative models that utilize an iterative denoising process to progressively sample from a learned data distribution, which actually is the reverse of a forward diffusion process. They have demonstrated impressive fidelity for text-conditioned image generation [53] and outperformed state-of-the-art GANs on ImageNet [16]. Despite the superior practical results, diffusion models suffer from a plodding sampling speed, *e.g.,* over hours to generate 50k CIFAR-sized images [59]. To accelerate the sampling from diffusion models, researchers propose to skip a part of the sampling steps by reframing the reverse chain [59, 40, 39], or distill the trained diffusion model into a faster one [47, 56]. Plus, the forward and backward processes in diffusion models can be formulated as stochastic differential equations [62], bridging diffusion models with Neural ODEs [13] in implicit deep learning. However, the community still lacks insights into the connection between DEQ and diffusion models, where we build our work to investigate this.

Model inversion Model inversion gives insights into the latent space of a generative model, as an inability of a generative model to correctly reconstruct an image from its latent code is indicative of its inability to model all the attributes of image correctly. Further, the ability to manipulate the latent codes to edit high-level attributes of images finds applications in many tasks like semantic image manipulation [67, 1], super resolution [12, 42], in-painting [15], compressed sensing [10], *etc.* For diffusion models like DDPM [29], the generative process is stochastic, which can make model inversion very challenging. Many existing works based on diffusion models [48, 15, 62, 31, 35] edit images or solve inverse problems without requiring full model inversion. Instead, they do so by utilizing existing understanding of diffusion models as presented in some recent works [62, 32, 34]. Diffusion models have been widely applied to conditional image generation [14, 15, 50, 55, 31, 57, 35, 49]. Chung et al. [15] propose a method to reduce the number of steps in reverse conditional diffusion process through better initialization, based on the idea of contraction theory of stochastic differential equations. Our proposed method is orthogonal to this work; we explicitly model DDIM as a joint, multi-variate fixed point system, and leverage black-box root solvers to solve for the fixed point.

7 Conclusion

We propose an approach to elegantly unify diffusion models and deep equilibrium models. We model the entire sampling chain of the denoising diffusion implicit model (DDIM) as a joint, multi-variate (deep) equilibrium model. This setup replaces the traditional sequential sampling process with a parallel one, thereby enabling us to enjoy speedup obtained from multiple GPUs. Further, we can leverage inexact gradients to optimize the entire sampling chain quickly, which results in significant gains in model inversion. We demonstrate the benefits of this approach on 1) single-shot image generation, where we were able to obtain FID scores on par with or slightly better than those of DDIM; and 2) model inversion, where we achieved faster convergence. It is possible to further speedup the sampling process by training a DEQ model to predict the noise at a particular timestep of the diffusion chain. We can jointly optimize the noise prediction network, and the latent variables of the diffusion chain, which we leave as future work.

References

[1] Rameen Abdal, Peihao Zhu, Niloy J Mitra, and Peter Wonka. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *ACM Transactions on Graphics (TOG)*, 40(3):1–21, 2021. (Cited on 2, 9)

[2] Brandon Amos. Tutorial on amortized optimization for learning to optimize over continuous domains. *arXiv preprint arXiv:2202.00665*, 2022. (Cited on 8)

[3] Brandon Amos and J. Zico Kolter. OptNet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning (ICML)*, 2017. (Cited on 8)

[4] Donald G Anderson. Iterative procedures for nonlinear integral equations. *Journal of the ACM (JACM)*, 1965. (Cited on 3, 4, 5, 8, 16, 20)

[5] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. Deep equilibrium models. *Neural Information Processing Systems (NeurIPS)*, 2019. (Cited on 3, 6, 8, 18)

[6] Shaojie Bai, Vladlen Koltun, and J Zico Kolter. Multiscale deep equilibrium models. *Neural Information Processing Systems (NeurIPS)*, 2020. (Cited on 8)

[7] Shaojie Bai, Vladlen Koltun, and J Zico Kolter. Stabilizing equilibrium models by jacobian regularization. *arXiv preprint arXiv:2106.14342*, 2021. (Cited on 6, 8)

[8] Shaojie Bai, Zhengyang Geng, Yash Savani, and J Zico Kolter. Deep equilibrium optical flow estimation. *arXiv preprint arXiv:2204.08442*, 2022. (Cited on 6, 8)

[9] Shaojie Bai, Vladlen Koltun, and J Zico Kolter. Neural deep equilibrium solvers. In *International Conference on Learning Representations*, 2022. (Cited on 8)

[10] Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G Dimakis. Compressed sensing using generative models. In *International Conference on Machine Learning*, pages 537–546. PMLR, 2017. (Cited on 9)

[11] Charles G Broyden. A class of methods for solving nonlinear simultaneous equations. *Mathematics of computation*, 1965. (Cited on 3, 8, 18)

[12] Kelvin CK Chan, Xintao Wang, Xiangyu Xu, Jinwei Gu, and Chen Change Loy. Glean: Generative latent bank for large-factor image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14245–14254, 2021. (Cited on 9)

[13] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Neural Information Processing Systems (NeurIPS)*, 2018. (Cited on 8, 9)

[14] Jooyoung Choi, Sungwon Kim, Yonghyun Jeong, Youngjune Gwon, and Sungroh Yoon. Ilvr: Conditioning method for denoising diffusion probabilistic models. *arXiv preprint arXiv:2108.02938*, 2021. (Cited on 9)

[15] Hyungjin Chung, Byeongsu Sim, and Jong Chul Ye. Come-closer-diffuse-faster: Accelerating conditional diffusion models for inverse problems through stochastic contraction. *arXiv preprint arXiv:2112.05146*, 2021. (Cited on 9)

[16] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34, 2021. (Cited on 1, 9)

[17] Josip Djolonga and Andreas Krause. Differentiable learning of submodular models. *Advances in Neural Information Processing Systems*, 30, 2017. (Cited on 8)

[18] Priya L. Donti, David Rolnick, and J Zico Kolter. DC3: A learning method for optimization with hard constraints. In *International Conference on Learning Representations (ICLR)*, 2021. (Cited on 8)

[19] Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural ODEs. In *Neural Information Processing Systems (NeurIPS)*, 2019. (Cited on 8)

[20] Laurent El Ghaoui, Fangda Gu, Bertrand Travacca, and Armin Askari. Implicit deep learning. *arXiv:1908.06315*, 2019. (Cited on 8)

[21] Thorsten Falk, Dominic Mai, Robert Bensch, Özgün Çiçek, Ahmed Abdulkadir, Yassine Marrakchi, Anton Böhm, Jan Deubner, Zoe Jäckel, Katharina Seiwald, et al. U-net: deep learning for cell counting, detection, and morphometry. *Nature methods*, 16(1):67–70, 2019. (Cited on 16)

[22] Samy Wu Fung, Howard Heaton, Qiuwei Li, Daniel McKenzie, Stanley Osher, and Wotao Yin. Fixed point networks: Implicit depth models with jacobian-free backprop. *arXiv e-prints*, pages arXiv–2103, 2021. (Cited on 6)

[23] Zhengyang Geng, Meng-Hao Guo, Hongxu Chen, Xia Li, Ke Wei, and Zhouchen Lin. Is attention better than matrix decomposition? In *International Conference on Learning Representations (ICLR)*, 2021. (Cited on 6, 8)

[24] Zhengyang Geng, Xin-Yu Zhang, Shaojie Bai, Yisen Wang, and Zhouchen Lin. On training implicit models. *Neural Information Processing Systems (NeurIPS)*, 2021. (Cited on 6, 8, 16)

[25] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. (Cited on 3)

[26] Albert Gu, Karan Goel, and Christopher Re. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations (ICLR)*, 2022. (Cited on 8)

[27] Fangda Gu, Heng Chang, Wenwu Zhu, Somayeh Sojoudi, and Laurent El Ghaoui. Implicit Graph Neural Networks. In *Neural Information Processing Systems (NeurIPS)*, pages 11984–11995, 2020. (Cited on 8)

[28] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. (Cited on 7)

[29] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Neural Information Processing Systems (NeurIPS)*, 2020. (Cited on 2, 3, 6, 9, 16)

[30] Thibaut Issenhuth, Ugo Tanielian, Jérémie Mary, and David Picard. Edibert, a generative model for image editing. *arXiv preprint arXiv:2111.15264*, 2021. (Cited on 2)

[31] Ajil Jalal, Marius Arvinte, Giannis Daras, Eric Price, Alexandros G Dimakis, and Jon Tamir. Robust compressed sensing mri with deep generative priors. *Advances in Neural Information Processing Systems*, 34:14938–14954, 2021. (Cited on 9)

[32] Zahra Kadkhodaie and Eero P Simoncelli. Solving linear inverse problems using the prior implicit in a denoiser. *arXiv preprint arXiv:2007.13640*, 2020. (Cited on 9)

[33] Kenji Kawaguchi. On the Theory of Implicit Deep Learning: Global Convergence with Implicit Layers. In *International Conference on Learning Representations (ICLR)*, 2020. (Cited on 8)

[34] Bahjat Kawar, Gregory Vaksman, and Michael Elad. Snips: Solving noisy inverse problems stochastically. *Advances in Neural Information Processing Systems*, 34:21757–21769, 2021. (Cited on 9)

[35] Gwanghyun Kim and Jong Chul Ye. Diffusionclip: Text-guided image manipulation using diffusion models. *arXiv preprint arXiv:2110.02711*, 2021. (Cited on 9)

[36] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. (Cited on 16)

[37] Diederik P Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *arXiv preprint arXiv:2107.00630*, 2021. (Cited on 1)

[38] J. Zico Kolter, David Duvenaud, and Matthew Johnson. Deep implicit layers tutorial - neural ODEs, deep equilibirum models, and beyond. *Neural Information Processing Systems Tutorial*, 2020. (Cited on 8, 16)

[39] Zhifeng Kong and Wei Ping. On fast sampling of diffusion probabilistic models. *arXiv preprint arXiv:2106.00132*, 2021. (Cited on 1, 9)

[40] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. In *International Conference on Learning Representations (ICLR)*, 2021. (Cited on 9)

[41] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009. (Cited on 2, 6)

[42] Haoying Li, Yifan Yang, Meng Chang, Shiqi Chen, Huajun Feng, Zhihai Xu, Qi Li, and Yueting Chen. Srdiff: Single image super-resolution with diffusion probabilistic models. *Neurocomputing*, 2022. (Cited on 9)

[43] Huan Ling, Karsten Kreis, Daiqing Li, Seung Wook Kim, Antonio Torralba, and Sanja Fidler. Editgan: High-precision semantic image editing. *Advances in Neural Information Processing Systems*, 34:16331–16345, 2021. (Cited on 2)

[44] Juncheng Liu, Kenji Kawaguchi, Bryan Hooi, Yiwei Wang, and Xiaokui Xiao. EIGNN: Efficient infinite-depth graph neural networks. In *Neural Information Processing Systems (NeurIPS)*, 2021. (Cited on 8)

[45] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015. (Cited on 2, 6)

[46] Cheng Lu, Jianfei Chen, Chongxuan Li, Qiuhao Wang, and Jun Zhu. Implicit normalizing flows. In *International Conference on Learning Representations (ICLR)*, 2021. (Cited on 8)

[47] Eric Luhman and Troy Luhman. Knowledge distillation in iterative generative models for improved sampling speed. *ArXiv*, abs/2101.02388, 2021. (Cited on 1, 9)

[48] Chenlin Meng, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021. (Cited on 9)

[49] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021. (Cited on 9)

[50] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021. (Cited on 1, 9)

[51] Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Anima Anandkumar. Diffusion models for adversarial purification. *arXiv preprint arXiv:2205.07460*, 2022. (Cited on 2)

[52] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. (Cited on 16)

[53] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022. (Cited on 9)

[54] Ardavan Saeedi, Matthew Hoffman, Stephen DiVerdi, Asma Ghandeharioun, Matthew Johnson, and Ryan Adams. Multimodal prediction and personalization of photo edits with deep generative models. In *International Conference on Artificial Intelligence and Statistics*, pages 1309–1317. PMLR, 2018. (Cited on 2)

[55] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *arXiv preprint arXiv:2104.07636*, 2021. (Cited on 9)

[56] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations (ICLR)*, 2022. (Cited on 1, 9)

[57] Hiroshi Sasaki, Chris G Willcocks, and Toby P Breckon. Unit-ddpm: Unpaired image translation with denoising diffusion probabilistic models. *arXiv preprint arXiv:2104.05358*, 2021. (Cited on 9)

[58] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning (ICML)*, 2015. (Cited on 2, 3, 9)

[59] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. (Cited on 1, 2, 3, 4, 6, 9, 16, 20, 22)

[60] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019. (Cited on 1, 9)

[61] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021. (Cited on 8)

[62] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021. (Cited on 9)

[63] Po-Wei Wang, Priya Donti, Bryan Wilder, and Zico Kolter. Satnet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver. In *International Conference on Machine Learning (ICML)*, 2019. (Cited on 8)

[64] Tiancai Wang, Xiangyu Zhang, and Jian Sun. Implicit Feature Pyramid Network for Object Detection. *arXiv preprint arXiv:2012.13563*, 2020. (Cited on 8)

[65] Ezra Winston and J. Zico Kolter. Monotone operator equilibrium networks. In *Neural Information Processing Systems (NeurIPS)*, 2020. (Cited on 8)

[66] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. LSUN: construction of a large-scale image dataset using deep learning with humans in the loop. *CoRR*, abs/1506.03365, 2015. (Cited on 6)

[67] Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. In-domain gan inversion for real image editing. In *European conference on computer vision*, pages 592–608. Springer, 2020. (Cited on 2, 9)

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes]
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]