Unsupervised Representation Learning from Pre-trained Diffusion Probabilistic Models

Anonymous Author(s)
Affiliation
Address
email

Abstract

Diffusion Probabilistic Models (DPMs) have shown a powerful capacity of generating high-quality image samples. Recently, diffusion autoencoders (Diff-AE) explore DPMs for representation learning via autoencoding and succeed in various downstream tasks. Their key idea is to jointly train an encoder for discovering meaningful representations from images and a conditional DPM as the decoder for image reconstruction. Considering that training DPMs will take a long time and there have already been many pre-trained unconditional DPMs, we aim to adapt these pre-trained models for representation learning also via autoencoding, but with less training times and better performance than Diff-AE. We find that the key factor that pre-trained DPMs cannot reconstruct image from latent variable is the information loss of forward diffusion process, which causes a gap between the predicted posterior mean by pre-trained DPMs and the true posterior mean. Inspired by the classifier-guided sampling method, we employ an encoder to learn meaningful representations from images and a gradient estimator to directly model the mean shift according to the learned representations to fill the posterior mean gap for image reconstruction. By further leveraging the knowledges of pre-trained DPMs and redesigning the weighting scheme of training objective, our method can learn richer representations from images more efficiently. Extensive experiments show that our method outperforms Diff-AE and enables some added tasks and features. We will make the code publicly available shortly.

1 Introduction

Deep generative models such as Variational Autoencoders (VAEs) [16, 24], Generative Adversarial Networks (GANs) [8], autoregressive models [34, 32], normalizing flows [23, 14] and energy-based models [6, 29] have shown remarkable capacity to synthesize striking image samples. Recently, another kind of generative models, Diffusion Probabilistic Models (DPMs) [27, 9] are further developed and become popular for their easy and stable training procedure and state-of-the-art sample quality [5]. Although a large number of degrees of freedom in implementation, the DPMs discussed in this paper will refer exclusively to the diffusion models trained by the denoising method proposed in DDPMs [9].

As latent variable generative models, DPMs inherently yield latent variables through the forward process. However, these latent variables are only images spatially corrupted by different levels of noise, which lacks high-level semantic information. Diffusion autoencoders (Diff-AE) [21] explore DPMs for representation learning under the paradigm of autoencoders. Specifically, they employ an encoder for discovering meaningful representations from images and a conditional DPM as the

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

decoder for image reconstruction by treating the representations as condition. Diff-AE is competitive with the state-of-the-art model on image reconstruction and capable of various downstream tasks.

Considering that training DPMs will take a long time and there have already been many pre-trained unconditional DPMs, we assess the following question: *Can we adapt pre-trained unconditional DPMs for representation learning?* To investigate this, we find that there exists a gap between the predicted posterior mean by pre-trained DPMs and the true posterior mean, which stems from the information loss of forward process and becomes the key factor that pre-trained DPMs cannot reconstruct image from latent variable ($\boldsymbol{x}_T$). Inspired by the classifier-guided sampling method [5], we employ an encoder to learn representations from images and a gradient estimator to directly model the mean shift according to the learned representations to fill the posterior mean gap for image reconstruction. In this way the encoder is encouraged to learn meaningful representations to retrieve the lost information. Furthermore, we find that the mean shift in different time stage contains different levels of information, so that we redesign a weighting scheme for training objective to encourage the model to learn rich representations efficiently. We also reuse a part of network of pre-trained DPMs to accelerate the convergence of our model. Following Diff-AE, we perform similar experiments and show that our method only needs aboud $\frac{1}{5}$ of the representation-relevant training times compared with Diff-AE but still outperforms it. Furthermore, our method also enables some added tasks and features.

2 Background

2.1 Denoising Diffusion Probabilistic Models

DDPMs [9] employ a forward process that starts from the data distribution $q(\boldsymbol{x}_0)$ and sequentially corrupts it to $\mathcal{N}(\mathbf{0}, \mathbf{I})$ with Markov diffusion kernels $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$ defined by a fixed variance schedule $\{\beta_t\}_{t=1}^T$ for T steps. One can express this process by:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}\,,\, \beta_t\mathbf{I}) \qquad q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) = \prod_{t=1}^{T} q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})\,, \tag{1}$$

where $\{\boldsymbol{x}_t\}_{t=1}^T$ are sequential latents of DDPMs. According to the rule of the sum of normally distributed random variables, we can directly sample $\boldsymbol{x}_t$ from $\boldsymbol{x}_0$ for arbitrary t with $q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0\,,\, (1-\bar{\alpha}_t)\mathbf{I})$, where $\alpha_t = 1-\beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^t \alpha_i$.

The reverse (generative) process is defined as another Markov chain parameterized by θ to describe the same but reverse process, denoising an arbitrary Gaussian noise to a clean data sample:

$$p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \mathcal{N}(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t)\,,\, \boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t, t)) \qquad p_\theta(\boldsymbol{x}_{0:T}) = p(\boldsymbol{x}_T)\prod_{t=1}^{T} p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)\,, \tag{2}$$

where $p(\boldsymbol{x}_T) = \mathcal{N}(\boldsymbol{x}_T; \mathbf{0}, \mathbf{I})$. It employs $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ of Gaussian form because the reversal of the diffusion process has the identical functional form as the forward process when β_t is small [27]. The generative distribution can be represented as $p_\theta(\boldsymbol{x}_0) = \int p_\theta(\boldsymbol{x}_{0:T})d\boldsymbol{x}_{1:T}$.

Training is performed to maximize the model log likelihood $\int q(\boldsymbol{x}_0)\log p_\theta(\boldsymbol{x}_0)d\boldsymbol{x}_0$ by minimizing the variational upper bound of negative one. The final objective is simplified by some parameterization, approximation and weighting [9]:

$$\mathcal{L}_{simple}(\theta) = \mathbb{E}_{t,\boldsymbol{x}_0,\epsilon}\left[\left\|\epsilon - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)\right\|^2\right], \tag{3}$$

where $\boldsymbol{\epsilon}_\theta$ is a function approximator to predict ϵ from $\boldsymbol{x}_t$.

2.2 Denoising Diffusion Implicit Models

DDIMs [28] define a non-Markovian forward process that leads to the same training objective with DDPMs, but the corresponding reverse process can be much more flexible and faster to sample from.

Specifically, one can sample $\boldsymbol{x}_{t-1}$ from $\boldsymbol{x}_t$ using the $\boldsymbol{\epsilon}_\theta$ of some pre-trained DDPMs via:

$$\boldsymbol{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\left(\frac{\boldsymbol{x}_t - \sqrt{1-\bar{\alpha}_t}\cdot\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t,t)}{\sqrt{\bar{\alpha}_t}}\right) + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\cdot\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t,t) + \sigma_t\epsilon_t\,, \tag{4}$$

where $\epsilon_t \sim \mathcal{N}(\mathbf{0},\mathbf{I})$ and σ_t controls the stochasticity of forward process. When $\sigma_t = 0$, the generative process becomes deterministic, which is named as DDIMs.

2.3 Classifier-guided Sampling Method

A common practice to condition DDPMs is to directly incorporate conditions into the function approximator $\boldsymbol{\epsilon}_\theta$. Beyond that, [27, 30, 5] show that a pre-trained unconditional DDPM can be conditioned on class information using the gradient of a classifier. Specifically, one can train a classifier $p_\phi(\boldsymbol{y}|\boldsymbol{x}_t)$ on noisy data and use its gradient $\nabla_{\boldsymbol{x}_t}\log p_\phi(\boldsymbol{y}|\boldsymbol{x}_t)$ to guide the diffusion sampling process towards specified class y. The conditional reverse process can be approximated by a Gaussian similar to the unconditional one in Eq.(2), but with a shifted mean[1]:

$$p_{\theta,\phi}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y}) \approx \mathcal{N}(\boldsymbol{x}_{t-1};\boldsymbol{\mu}_\theta(\boldsymbol{x}_t,t) + \boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t,t)\cdot\nabla_{\boldsymbol{x}_t}\log p_\phi(\boldsymbol{y}|\boldsymbol{x}_t)\,,\ \boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t,t))\,. \tag{5}$$

For deterministic sampling methods like DDIMs, we can use score-based conditioning trick [30, 29] to define a new function approximator for conditional sampling:

$$\hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{x}_t,t) = \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t,t) - \sqrt{1-\bar{\alpha}_t}\cdot\nabla_{\boldsymbol{x}_t}\log p_\phi(\boldsymbol{y}|\boldsymbol{x}_t)\,. \tag{6}$$

More generally, any similarity estimator between noisy data and conditions can be applied for guided sampling, such as noisy-CLIP guidance [19, 18] that enables continuous conditions.

3 Method

3.1 Forward Process Posterior Mean Gap

Generally, one will train unconditional and conditional DPMs by learning $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \mathcal{N}(\boldsymbol{x}_{t-1};\boldsymbol{\mu}_\theta(\boldsymbol{x}_t,t),\boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t,t))$ and $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{y}) = \mathcal{N}(\boldsymbol{x}_{t-1};\boldsymbol{\mu}_\theta(\boldsymbol{x}_t,\boldsymbol{y},t),\boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t,\boldsymbol{y},t))$ to approximate the same forward process posterior $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_{t-1};\widetilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t,\boldsymbol{x}_0),\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t\mathbf{I})$. Here $\boldsymbol{y}$ is some condition that contains some prior knowledges of corresponding data, such as class label. Usually, under the same experimental settings, the conditional DPMs will have a lower optimized loss and be able to reconstruct the information of given $\boldsymbol{y}$ in samples. Furthermore, according to Eq.(5), the mean of conditional reverse process contains an extra shift item relative to that of unconditional one. These imply that there exists a gap between the predicted posterior mean by the unconditional DPMs ($\boldsymbol{\mu}_\theta(\boldsymbol{x}_t,t)$) and the true posterior mean ($\widetilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t,\boldsymbol{x}_0)$). The posterior mean gap is caused by the information loss of forward process so that the unconditional reverse process cannot accurately reconstruct the information in samples unless introducing some guidance (input condition or mean shift). Intuitively, this is due to $\widetilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t,\boldsymbol{x}_0)$ knows $\boldsymbol{x}_0$ while $\boldsymbol{\mu}_\theta(\boldsymbol{x}_t,t)$ not.

In theory, if $\boldsymbol{y}$ in Eq.(5) contains all information of $\boldsymbol{x}_0$, the mean shift will fill the posterior mean gap and guide the reverse process to reconstruct $\boldsymbol{x}_0$. On the other hand, if we replace $\boldsymbol{y}$ with our learned representations $\boldsymbol{z}$, we can train our autoencoder by modeling the mean shift according to $\boldsymbol{z}$ to fill the posterior mean gap for image reconstruction as much as possible, in this way the encoder learn meaningful representations to retrieve the lost information as much as possible. The more the posterior mean gap is filled, the more accurate the mean shift is, and the more information of $\boldsymbol{x}_0$ that $\boldsymbol{z}$ contains.

3.2 Unsupervised Representation Learning

Following the paradigm of autoencoders, we employ an encoder $\boldsymbol{z} = \boldsymbol{E}_\varphi(\boldsymbol{x}_0)$ for learning compact and meaningful representations from input images and adapt a pre-trained unconditional DPM $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \mathcal{N}(\boldsymbol{x}_{t-1};\boldsymbol{\mu}_\theta(\boldsymbol{x}_t,t),\boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t,t))$ as the decoder for image reconstruction.

[1] In [5], they use the gradient $\nabla_{\boldsymbol{x}}\log p_\phi(\boldsymbol{y}|\boldsymbol{x})$ at $\boldsymbol{x} = \boldsymbol{\mu}_\theta(\boldsymbol{x}_t,t)$ as mean shift during derivation, but use the one at $\boldsymbol{x} = \boldsymbol{x}_t$ during guided sampling. We take the same simplication in this paper.

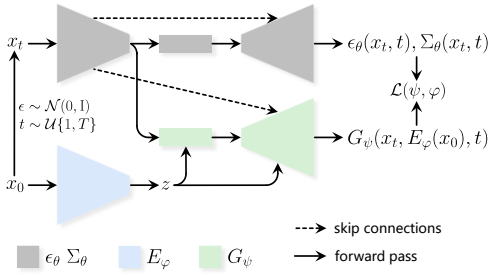

Figure 1: Network and data flow of our method. The gray part represents the pre-trained DPM, which is frozen during training.

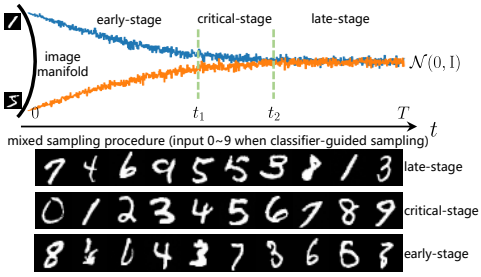

Figure 2: Investigations of the affects of mean shift for different time stage. We set $t_1 = 400$ and $t_2 = 600$ for MNIST [17].

Specifically, we employ a gradient estimator $\boldsymbol{G}_\psi(\boldsymbol{x}_t, \boldsymbol{z}, t)$ that approximates $\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{z}|\boldsymbol{x}_t)$ to fill the posterior mean gap between $\boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t)$ and $\widetilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t, \boldsymbol{x}_0)$. Here $p(\boldsymbol{z}|\boldsymbol{x}_t)$ is some implicit classifier (or similarity estimator described in Section 2.3) that we will not use explicitly. Then we can assemble a conditional DPM as $p_{\theta,\psi}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t) + \boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t, t) \cdot \boldsymbol{G}_\psi(\boldsymbol{x}_t, \boldsymbol{z}, t)\,,\ \boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t, t))$ and train it like a normal DPM with the following derived objective (assuming the ϵ-prediction parameterization is adopted):

$$\mathcal{L}(\psi, \varphi) = \mathbb{E}_{t,\boldsymbol{x}_0,\epsilon}\left[\lambda_t \left\|\epsilon - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t) + \frac{\sqrt{\alpha_t}\sqrt{1-\bar{\alpha}_t}}{\beta_t} \cdot \boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t, t) \cdot \boldsymbol{G}_\psi(\boldsymbol{x}_t, \boldsymbol{E}_\varphi(\boldsymbol{x}_0), t)\right\|^2\right], \quad (7)$$

where $\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$ and λ_t is a new weighting scheme that we will discuss in Section 3.4. Note that we use pre-trained DPMs so that θ are frozen during optimization. Usually we set $\boldsymbol{\Sigma}_\theta = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t \mathbf{I}$ to untrained time-dependent constants.

With trained $\boldsymbol{G}_\psi(\boldsymbol{x}_t, \boldsymbol{z}, t)$, we can treat it as the gradient of an optimal classifier $p(\boldsymbol{z}|\boldsymbol{x}_t)$ and use the classifier-guided sampling method in Eq.(5) for DDPM sampling or use the modified function approximator $\hat{\boldsymbol{\epsilon}}_\theta$ in Eq.(6) for DDIM sampling, based on pre-trained $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)$.

Except the semantic latent code $\boldsymbol{z}$, we can infer a stochastic latent code $\boldsymbol{x}_T$ by running the deterministic generative process of DDIMs in reverse:

$$\boldsymbol{x}_{t+1} = \sqrt{\bar{\alpha}_{t+1}}\left(\frac{\boldsymbol{x}_t - \sqrt{1-\bar{\alpha}_t} \cdot \hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{x}_t, t)}{\sqrt{\bar{\alpha}_t}}\right) + \sqrt{1-\bar{\alpha}_{t+1}} \cdot \hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{x}_t, t)\,. \quad (8)$$

The stochastic latent code can capture minor details from input images [21]. It is optional but helpful to near-exact reconstruction and real-image manipulation for reconstructing minor details when using DDIM sampling.

We also train a normal DPM $p_\omega(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t)$ to model the learned semantic latent space, same with that in Diff-AE [21]. With a trained latent DPM, we can sample $\boldsymbol{z}$ from it and help pre-trained DPMs to achieve faster and better unconditional sampling.

3.3 Network Design

Figure 1 shows the network and data flow of our method. For encoder $\boldsymbol{E}_\varphi$, we simply use stacked convolution layers and a linear layer to map $\boldsymbol{x}_0$ to $\boldsymbol{z}$. For gradient estimator $\boldsymbol{G}_\psi$, we use the U-Net [25] similar to the function approximator of pre-trained DPM. Considering that $\boldsymbol{\epsilon}_\theta$ also takes $\boldsymbol{x}_t$ and t as input, we can further leverage the knowledges of pre-trained DPM by reusing its trained encoder part and time embedding layer. Then we employ new middle blocks, decoder part and output blocks of U-Net and incorporate $\boldsymbol{z}$ into them. Following [5], we extend Group Normalization [37] by applying scaling & shifting twice on normalized feature maps $\boldsymbol{h}$:

$$\mathrm{AdaGN}(\boldsymbol{h}, t, \boldsymbol{z}) = \boldsymbol{z}_s(t_s \mathrm{GroupNorm}(\boldsymbol{h}) + t_b) + \boldsymbol{z}_b\,, \quad (9)$$

where $[\boldsymbol{t}_s, \boldsymbol{t}_b]$ and $[\boldsymbol{z}_s, \boldsymbol{z}_b]$ are obtained from a linear projection of t and $\boldsymbol{z}$, respectively. Note that we still use skip connections from reused encoder to new decoder. In this way, $\boldsymbol{G}_\psi$ is determined by pre-trained DPM and can be universally applied on different architectures.

3.4 Weighting scheme Redesign

P2-weighting [4] shows that the weighting scheme of objective function greatly affects the performance of DPMs. We investigate similar phenomenon for representation learning in Figure 2.

Specifically, we train an unconditional DPM and a classifier on MNIST [17], and divide the diffusion forward process into three stages: early-stage between 0 and t_1, critical-stage between t_1 and t_2 and late-stage between t_2 and T, as shown in the top row. Then we design a mixed sampling procedure that employs unconditional sampling but switches to classifier-guided sampling during the specified stage. The bottom three rows show the samples generated by three different mixed sampling procedures, where each row only employs classifier-guided sampling for the right stage. As we can see, only the samples guided by the classifier during critical-stage match the input class labels. We can conclude that the mean shift during critical-stage contains more crucial information to reconstruct the input class label in samples than the other two stages. From the view of diffusion trajectories, the sampling trajectories are separated from each other during critical-stage and they need the mean shift to guide them towards specified direction, otherwise it will be determined by the stochasticity of Langevin dynamics. Therefore, we opt to down-weight the objective function for the t in early-stage and late-stage to encourage the model to learn rich representations from critical-stage. Inspired by P2-weighting [4], we redesign a weighting scheme of objective function (λ_t in Eq.(7)) in terms of signal-to-noise ratio [15] ($\mathrm{SNR}(t) = \frac{\bar{\alpha}_t}{1-\bar{\alpha}_t}$):

$$\lambda_t = \left(\frac{1}{1+\mathrm{SNR}(t)}\right)^{\gamma} \cdot \left(\frac{\mathrm{SNR}(t)}{1+\mathrm{SNR}(t)}\right)^{1-\gamma}, \tag{10}$$

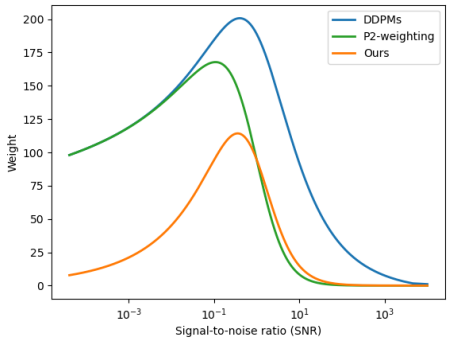

Figure 3: Weighting schemes of different DPMs relative to the true variational lower bound loss. (linear noise schedule)

where the first item is for early-stage and the second one is for late-stage. γ is a hyperparameter that balances the strength of down-weighting between two items. Empirically we set $\gamma = 0.25$. Figure 3 shows the weighting schemes of different DPMs relative to the true variational lower bound loss. Compared with other DPMs, our weighting scheme down-weights the loss for both low and high SNR. We find this weighting scheme is indispensable, otherwise the model will not converge.

4 Experiments

To compare our method with Diff-AE [21], we follow their experiments under the same settings. Furthermore, we also show that our method enables some added tasks and features. For fair comparison, we use the baseline DPMs provided by Diff-AE in [2] as our pre-trained models (also as one of our baselines). For brevity, we use the notation such as "FFHQ128-130M-z512-22M" to name our model, which means that we use a baseline DPM pre-trained with 130M images and adapt it for representation learning by our method with 22M images, on 128×128 FFHQ dataset, with the latent $\boldsymbol{z}$ of 512-d. We put all implementation details in Appendix A and additional samples of following experiments in Appendix B.

4.1 Learned Mean Shift Fills Posterior Mean Gap

Similar to Diff-AE, we train a model of "FFHQ128-130M-z512-22M" on FFHQ [12] and evaluate it on unseen CelebA-HQ [11]. We first show that our learned mean shift can fill the posterior mean gap with qualitative and quantitative results in Figure 4. Specifically, we sample some $\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$ for different t and predict $\boldsymbol{x}_0$ from $\boldsymbol{x}_t$ by denoising them for only one step, using

[2] https://github.com/phizaz/diffae

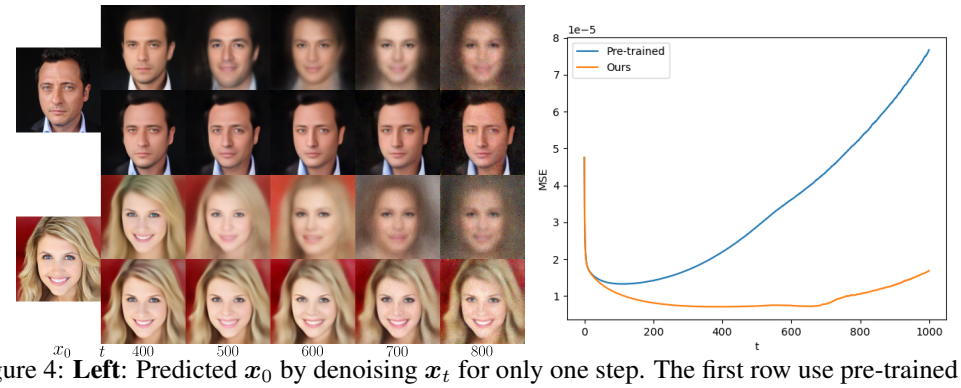

Figure 4: **Left**: Predicted $\boldsymbol{x}_0$ by denoising $\boldsymbol{x}_t$ for only one step. The first row use pre-trained DPM and the second row use our method. **Right**: Average posterior mean gap for all steps.

Figure 5: Autoencoding reconstruction examples generated by "FFHQ128-130M-z512-22M" on CelebA-HQ with different sampling methods. Each row corresponds to an exmaple.

pre-trained DPM and our method respectively. As we can see in the figure (left), even for large t, our method can predict accurate noise from $\boldsymbol{x}_t$ and reconstruct plausible images, which shows that our method indeed fills the posterior mean gap by modeling mean shift and learns meaningful representations from images to retrieve the lost information of forward process.

Furthermore, we randomly select 1000 images from CelebA-HQ and calculate their average posterior mean gap for all steps using pre-trained DPM: $\|\widetilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t, \boldsymbol{x}_0) - \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t)\|^2$ and our method: $\|\widetilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t, \boldsymbol{x}_0) - (\boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t) + \boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t, t) \cdot \boldsymbol{G}_\psi(\boldsymbol{x}_t, \boldsymbol{E}_\varphi(\boldsymbol{x}_0), t))\|^2$, shown in the figure (right). As we can see, our method models the mean shift that greatly fills the posterior mean gap.

4.2 Autoencoding Reconstruction

We use "FFHQ128-130M-z512-22M" to compute the semantic latent code $\boldsymbol{z} = \boldsymbol{E}_\varphi(\boldsymbol{x}_0)$ for CelebA-HQ images and run generative process of DDIM and DDPM respectively. We present some autoencoding reconstruction examples in Figure 5. As we can see, both methods generate samples with similar contents to the input, which shows that $\boldsymbol{E}_\varphi$ captures the semantic information analogous to the class label and $\boldsymbol{G}_\psi$ learns the mean shift analogous to the gradient of a classifier. Some stochastic variations occur in minor details of hair, eye and skin when introducing stochasticity. Due to the similar performance between DDPM and DDIM with random $\boldsymbol{x}_T$, we will always use DDIM sampling method in later experiments. We can get a near-exact reconstruction if we use the stochastic latent code inferred from aforementioned ODE, which further proves that the stochastic latent code controls the local details.

To further evaluate the autoencoding reconstruction quality of our method, we conduct the same quantitative experiments with Diff-AE. Specifically, we use "FFHQ128-130M-z512-22M" to encode-and-reconstruct all 30k images of CelebA-HQ and evaluate the reconstruction quality with SSIM [36], LPIPS [39] and MSE. We use the same baselines described in [21], and the results are shown in Table 1. We can see that our method is competitive with the state-of-the-art NVAE even with

Table 1: Autoencoding reconstruction quality of "FFHQ128-130M-z512-22M" on CelebA-HQ.

Model	**Latent dim**	**SSIM** $\uparrow$	**LPIPS** $\downarrow$	**MSE** $\downarrow$
StyleGAN2 ($\mathcal{W}$ inversion) [13]	512	0.677	0.168	0.016
StyleGAN2 ($\mathcal{W}$+ inversion) [1, 2]	7,168	0.827	0.114	0.006
VQ-GAN [7]	65,536	0.782	0.109	3.61e-3
VQ-VAE2 [22]	327,680	0.947	0.012	4.87e-4
NVAE [31]	6,005,760	0.984	**0.001**	**4.85e-5**
Diff-AE @130M (T=100, random $\boldsymbol{x}_T$) [21]	512	0.677	0.073	0.007
Ours (T=100, random $\boldsymbol{x}_T$)	512	0.691	0.110	0.005
DDIM @130M (T=100) [28]	49,152	0.917	0.063	0.002
Diff-AE @130M (T=100, inferred $\boldsymbol{x}_T$) [21]	49,664	0.991	0.011	6.07e-5
Ours (T=100, inferred $\boldsymbol{x}_T$)	49,664	**0.993**	0.007	5.37e-5

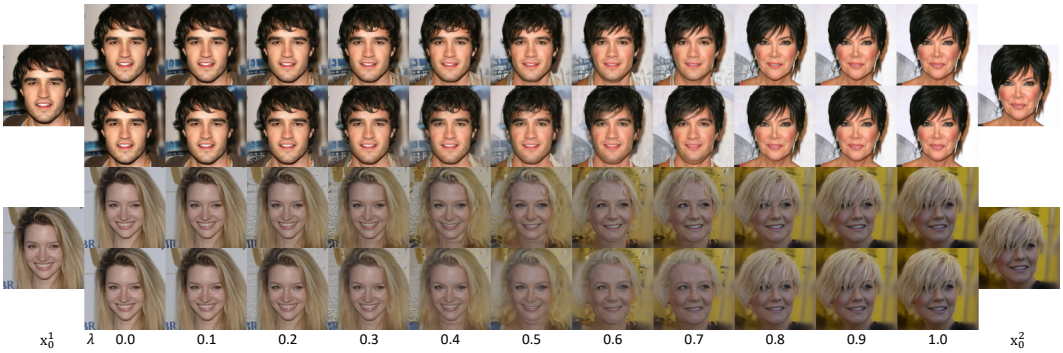

Figure 6: Interpolation examples generated by "FFHQ128-130M-z512-22M" on CelebA-HQ. For each example, the first row use the guidance of $\boldsymbol{G}_\psi\big(\boldsymbol{x}_t, Lerp(\boldsymbol{z}^1, \boldsymbol{z}^2; \lambda), t\big)$ and the second row use the guidance of $Lerp\big(\boldsymbol{G}_\psi(\boldsymbol{x}_t, \boldsymbol{z}^1, t), \boldsymbol{G}_\psi(\boldsymbol{x}_t, \boldsymbol{z}^2, t); \lambda\big)$.

much less latent dimensionality and also outperforms Diff-AE in all metrics except the LPIPS for random $\boldsymbol{x}_T$. Moreover, our method only needs aboud $\frac{1}{5}$ of the representation-relevant training times compared with Diff-AE, which shows that our method can learn richer representations from images more efficiently.

4.3 Interpolation of Semantic Latent Codes and Trajectories

Given two samples $\boldsymbol{x}_0^1$ and $\boldsymbol{x}_0^2$ from CelebA-HQ, we first use "FFHQ128-130M-z512-22M" to encode them into $(\boldsymbol{z}^1, \boldsymbol{x}_T^1)$ and $(\boldsymbol{z}^2, \boldsymbol{x}_T^2)$ and then run the generative process of DDIM starting from $Slerp(\boldsymbol{x}_T^1, \boldsymbol{x}_T^2; \lambda)$ under the guidance of $\boldsymbol{G}_\psi\big(\boldsymbol{x}_t, Lerp(\boldsymbol{z}^1, \boldsymbol{z}^2; \lambda), t\big)$ with 100 steps, expecting smooth transitions along λ. Moreover, from the view of the diffusion trajectories, our method generates desired samples by shifting the unconditional sampling trajectories towards the spatial direction predicted by $\boldsymbol{G}_\psi(\boldsymbol{x}_t, \boldsymbol{z}, t)$. This enables our method to directly interpolate between two different sampling trajectories. Intuitively, the spatial direction predicted by the linear interpolation of two semantic latent codes, $\boldsymbol{G}_\psi\big(\boldsymbol{x}_t, Lerp(\boldsymbol{z}^1, \boldsymbol{z}^2; \lambda), t\big)$, should be equivalent to the linear interpolation of two spatial directions predicted by respective semantic latent code, $Lerp\big(\boldsymbol{G}_\psi(\boldsymbol{x}_t, \boldsymbol{z}^1, t), \boldsymbol{G}_\psi(\boldsymbol{x}_t, \boldsymbol{z}^2, t); \lambda\big)$. We present some examples of these two kinds of interpolation methods in Figure 6. As we can see, both methods generate similar samples that smoothly transition from one endpoint to the other, which means that $\boldsymbol{G}_\psi\big(\boldsymbol{x}_t, Lerp(\boldsymbol{z}^1, \boldsymbol{z}^2; \lambda), t\big) \approx Lerp\big(\boldsymbol{G}_\psi(\boldsymbol{x}_t, \boldsymbol{z}^1, t), \boldsymbol{G}_\psi(\boldsymbol{x}_t, \boldsymbol{z}^2, t); \lambda\big)$, so that $\boldsymbol{G}_\psi(\boldsymbol{x}_t, \boldsymbol{z}, t)$ can be seen as a function of $\boldsymbol{z}$ analogous to a linear map. The linearity guarantees a meaningful semantic latent space that represents the semantic spatial change of image by a linear change of latent code.

4.4 Attribute Manipulation

We can further explore the semantic latent space in a supervised way. To show this, we conduct attribute manipulation experiments based on "FFHQ128-130M-z512-22M". Specifically, we first encode an image to its semantic latent code, then move it along the learned direction and finally

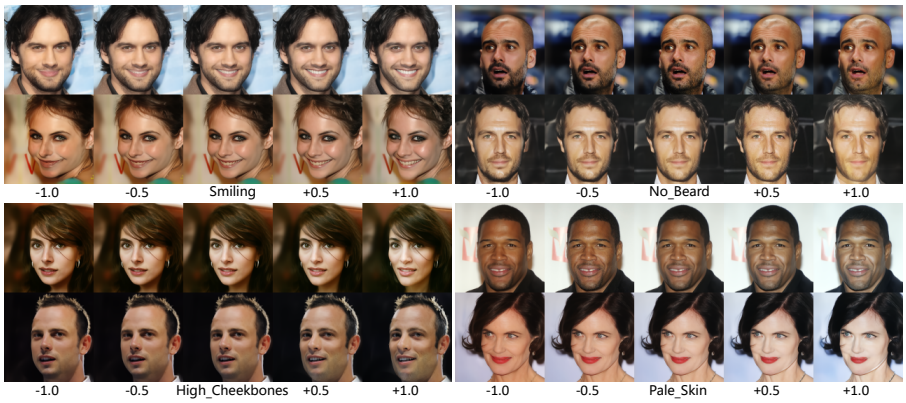

Figure 7: Attribute manipulation examples generated by "FFHQ128-130M-z512-22M" on CelebA-HQ. For each example, we manipulate the input image (middle) by moving its semantic latent code along the direction of corresponding attribute found by trained linear classifiers with different scales.

decode it to the manipulated image. Similar to Diff-AE, we train a linear classifier to separate the semantic latent codes of the images with different attribute labels and use the normal vector of separating hyperplane (i.e. the weight of linear classifier) as the direction vector. We use the images and attribute labels from CelebA-HQ to train the linera classifier. We present some attribute manipulation examples in Figure 7. As we can see, our method succeed in manipulating images by moving their semantic latent codes along the direction of desired attribute with different scales. Like Diff-AE, our method can change attribute-relevant features while keeping other irrelevant details almost stationary if we use the inferred $\boldsymbol{x}_T$ of input image.

4.5 Few-shot Conditional Generation

Followig D2C [26], we train a model of "CelebA64-64M-z512-13M" on CelebA [11] and aim to achieve conditional sampling given a small number of labeled images. Specifically, we train a latent DPM $p_\omega(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t)$ on the semantic latent space and a latent classifier $p_\eta(y|\boldsymbol{z})$ with given labeled images. For binary scenario, the images are labeled by a binary class (100 samples, 50 for each class). For PU scenario, the images are either labeled positive or unlabeled (100 positively labeled and 10k unlabeled samples). Then we sample $\boldsymbol{z}$ from $p_\omega(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t)$ and accept it with the probability of $p_\eta(y|\boldsymbol{z})$. We use the accepted $\boldsymbol{z}$ to generate 5k samples for every class and compute the FID score between these samples and all images belonging to corresponding class in dataset. We compare our method with Diff-AE and D2C. We also use the naive approach that computes the FID score between the training images and the corresponding subset of images in dataset. Table 2 shows that our method achieves better FID scores than Diff-AE, which further proves the meaningfulness of our semantic latent space.

Table 2: FID scores for few-shot conditional generation using "CelebA64-64M-z512-13M".

Scenario	Classes	Ours	Diff-AE [21]	D2C [26]	Naive
Binary	Male	**10.99**	11.52	13.44	22.81
	Female	**6.65**	7.29	9.51	13.56
	Blond	**15.78**	16.10	17.61	22.48
	Non-Blond	8.78	**8.48**	8.94	1.00
PU	Male	**9.23**	9.54	16.39	22.81
	Female	**8.64**	9.21	12.21	13.56
	Blond	**6.51**	7.01	10.09	22.48
	Non-Blond	**6.89**	7.91	9.09	1.00

4.6 Improved Unconditional Sampling and Truncation-like Effect

Unlike Diff-AE that must take $\boldsymbol{z}$ as input for sampling, our method doesn't obstruct the unconditional sampling of pre-trained DPMs. However, our method can improve the sample quality and accelerate the sampling speed of pre-trained DPMs. Therefore, we can train latent DPMs on semantic latent space to sample $\boldsymbol{z}$ to improve the unconditional sample quality of pre-trained DPMs. We evaluate this method with "FFHQ128-130M-z512-22M", "Horse128-130M-z512-35M", "Bedroom128-120M-z512-25M" and "CelebA64-64M-z512-13M" by DDIM sampling method with different steps. For

Dataset	Model	FID			
		T=10	T=20	T=50	T=100
FFHQ	DDIM	31.87	20.53	15.82	11.95
	Diff-AE	21.95	18.10	13.14	10.55
	Ours	**19.38**	**15.35**	**11.37**	**10.27**
Horse	DDIM	25.24	14.41	7.98	5.93
	Diff-AE	**12.66**	9.21	7.12	5.27
	Ours	13.60	**8.89**	**6.81**	**4.96**
Bedroom	DDIM	14.07	9.29	7.31	5.88
	Diff-AE	10.79	8.42	**6.49**	5.32
	Ours	**10.28**	**7.63**	6.68	**5.27**
CelebA	DDIM	18.89	13.82	8.48	5.94
	Diff-AE	12.92	10.18	7.05	5.30
	Ours	**12.27**	**9.23**	**6.70**	**5.22**

Table 3: FID scores for improved unconditional sampling.

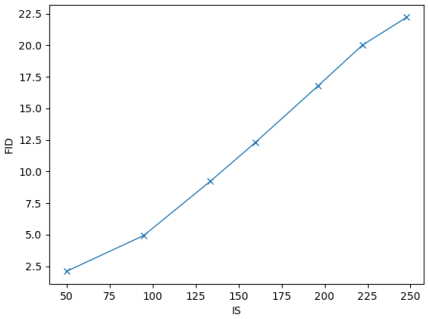

Figure 8: Truncation-like effect for "ImageNet64-51M-y-13M" by scaling $\boldsymbol{G}_\psi(\boldsymbol{x}_t, y, t)$ with 0.0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0 respectively

each dataset, we calculate the FID scores between 50k generated samples and 50k real images randomly selected from dataset. Table 3 shows that our method can significantly improve the sample quality of pre-trained DPMs and outperforms Diff-AE. Note that our method can be applied for any pre-trained DPMs as an auxiliary booster to improve their sample quality.

Inspired by [5, 10], we can obtain a truncation-like effect in DPMs by scaling the classifier guidance. We have shown that our $\boldsymbol{G}_\psi(\boldsymbol{x}_t, \boldsymbol{z}, t)$ simulates the gradient of some implicit classifier. Therefore we can discard our encoder and directly incorporate the class label into $\boldsymbol{G}_\psi(\boldsymbol{x}_t, y, t)$ to learn a mean shift by filling the posterior mean gap, then we can scale the mean shift for a truncation-like effect. We train a model of "ImageNet64-51M-y-13M" and use DDIM sampling method with 100 steps to generate 50k samples. Figure 8 shows the sample quality effects of sweeping over the scale.

4.7 Disentangled Representation Learning

Our method can also be applied for disentangled representation learning. Due to limited space, we put details in Appendix C.

5 Related Work

Unsupervised representation learning via generative modeling is a popular topic in computer vision. Latent variable generative models, such as GANs [8] and VAEs [16, 24], are a natural candidate for this, since they inherently involve a latent representation of the data they generate. For GANs, due to its lack of inference functionality, one have to extract the representations for any given real samples by extra techniques called GAN Inversion [38], which invert samples back into the latent space of trained GANs. Existing inversion methods [40, 20, 3, 1, 2, 35] either have limited reconstruction quality or need significantly higher computational cost. Autoencoder-based generative models such as VAEs explicitly learn representations for samples, but still face representation-generation trade-off challenges [33, 26]. VQ-VAE [33, 22] and D2C [26] overcome these problems by modeling latent variables post-hoc in different ways. Diffusion autoencoders (Diff-AE) [21] explore DPMs for representation learning via autoencoding. Specifically, they jointly train an encoder for discovering meaningful representations from images and a conditional DPM as the decoder for image reconstruction by treating the representations as condition. Diff-AE is competitive with the state-of-the-art model on image reconstruction and capable of various downstream tasks.

6 Conclusion

We have presented a method that adapts pre-trained DPMs for representation learning via autoencoding with less training times and better performance than Diff-AE [21]. By modeling the mean shift that can fill the posterior mean gap to retrieve the lost information of forward diffusion process, the encoder can learn meaningful representation from input images for reconstruction. Our method is more flexible than Diff-AE and enables some added tasks and features.

References

[1] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4432–4441, 2019.

[2] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan++: How to edit the embedded images? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8296–8305, 2020.

[3] David Bau, Jun-Yan Zhu, Jonas Wulff, William Peebles, Hendrik Strobelt, Bolei Zhou, and Antonio Torralba. Inverting layers of a large generator. In *ICLR Workshop*, volume 2, page 4, 2019.

[4] Jooyoung Choi, Jungbeom Lee, Chaehun Shin, Sungwon Kim, Hyunwoo Kim, and Sungroh Yoon. Perception prioritized training of diffusion models. *arXiv preprint arXiv:2204.00227*, 2022.

[5] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34, 2021.

[6] Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. *Advances in Neural Information Processing Systems*, 32, 2019.

[7] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12873–12883, 2021.

[8] 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.

[9] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arXiv:2006.11239*, 2020.

[10] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

[11] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[12] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[13] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020.

[14] Diederik P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *arXiv preprint arXiv:1807.03039*, 2018.

[15] Diederik P Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *arXiv preprint arXiv:2107.00630*, 2021.

[16] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[17] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[18] Xihui Liu, Dong Huk Park, Samaneh Azadi, Gong Zhang, Arman Chopikyan, Yuxiao Hu, Humphrey Shi, Anna Rohrbach, and Trevor Darrell. More control for free! image synthesis with semantic diffusion guidance. *arXiv preprint arXiv:2112.05744*, 2021.

[19] 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.

[20] Guim Perarnau, Joost Van De Weijer, Bogdan Raducanu, and Jose M Álvarez. Invertible conditional gans for image editing. *arXiv preprint arXiv:1611.06355*, 2016.

[21] Konpat Preechakul, Nattanat Chatthee, Suttisak Wizadwongsa, and Supasorn Suwajanakorn. Diffusion autoencoders: Toward a meaningful and decodable representation. *arXiv preprint arXiv:2111.15640*, 2021.

[22] Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019.

[23] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

[24] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR, 2014.

[25] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[26] Abhishek Sinha, Jiaming Song, Chenlin Meng, and Stefano Ermon. D2c: Diffusion-decoding models for few-shot conditional generation. *Advances in Neural Information Processing Systems*, 34, 2021.

[27] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

[28] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[29] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

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

[31] Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder. *Advances in Neural Information Processing Systems*, 33:19667–19679, 2020.

[32] Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. *Advances in neural information processing systems*, 29, 2016.

[33] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[34] Aaron Van Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *International Conference on Machine Learning*, pages 1747–1756. PMLR, 2016.

[35] Andrey Voynov and Artem Babenko. Unsupervised discovery of interpretable directions in the gan latent space. In *International conference on machine learning*, pages 9786–9796. PMLR, 2020.

[36] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pages 1398–1402. Ieee, 2003.

[37] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

[38] Weihao Xia, Yulun Zhang, Yujiu Yang, Jing-Hao Xue, Bolei Zhou, and Ming-Hsuan Yang. Gan inversion: A survey. *arXiv preprint arXiv:2101.05278*, 2021.

[39] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[40] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *European conference on computer vision*, pages 597–613. Springer, 2016.

Checklist

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? [No]
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (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? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
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)? [No] But we will release the code later.
   (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? [No]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (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]