DisCo-Diff: Enhancing Continuous Diffusion Models with Discrete Latents

Yilun Xu [*1 2] **Gabriele Corso** [2] **Tommi Jaakkola** [2] **Arash Vahdat** [1] **Karsten Kreis** [1]

Abstract

Diffusion models (DMs) have revolutionized generative learning. They utilize a diffusion process to encode data into a simple Gaussian distribution. However, encoding a complex, potentially multimodal data distribution into a single *continuous* Gaussian distribution arguably represents an unnecessarily challenging learning problem. We propose ***Dis**crete-**Co**ntinuous Latent Variable **Diff**usion Models (DisCo-Diff)* to simplify this task by introducing complementary *discrete* latent variables. We augment DMs with learnable discrete latents, inferred with an encoder, and train DM and encoder end-to-end. DisCo-Diff does not rely on pre-trained networks, making the framework universally applicable. The discrete latents significantly simplify learning the DM's complex noise-to-data mapping by reducing the curvature of the DM's generative ODE. An additional autoregressive transformer models the distribution of the discrete latents, a simple step because DisCo-Diff requires only few discrete variables with small codebooks. We validate DisCo-Diff on toy data, several image synthesis tasks as well as molecular docking, and find that introducing discrete latents consistently improves model performance. For example, DisCo-Diff achieves state-of-the-art FID scores on class-conditioned ImageNet-64/128 datasets with ODE sampler.

1. Introduction

Diffusion models (DMs) (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021) have recently led to breakthroughs for generative modeling in diverse domains. For instance, they can synthesize expressive high-resolution imagery (Saharia et al., 2022; Ramesh et al., 2022; Rombach et al., 2022; Balaji et al., 2022) or they can generate accurate molecular structures (Corso et al., 2023; Yim et al., 2023; Ingraham et al., 2023; Watson et al., 2023). DMs leverage a forward diffusion process that effectively encodes the training data in a simple, unimodal Gaussian prior distribution. Generation can be formulated either as a stochastic or, more conveniently, as a deterministic process that takes as input random noise from the Gaussian prior and transforms it into data through a generative ordinary differential equation (ODE) (Song et al., 2021). The Gaussian prior corresponds to the DM's *continuous latent variables*, where the data is uniquely encoded through the ODE-defined mapping.

However, realistic data distributions are typically high-dimensional, complex and often multimodal. Directly encoding such data into a single unimodal Gaussian distribution and learning a corresponding reverse noise-to-data mapping is challenging. The mapping, or generative ODE, necessarily needs to be highly complex, with strong curvature, and one may consider it unnatural to map an entire data distribution to a single Gaussian distribution. In practice, conditioning information, such as class labels or text prompts, often helps to simplify the complex mapping by offering the DM's denoiser additional cues for more accurate denoising. However, such conditioning information is typically of a semantic nature and, even given a class or text prompt, the mapping remains highly complex. For instance, in the case of images, even within a class we find images with vastly different styles and color patterns, which corresponds to large distances in pixel space.

Here, we propose ***Dis**crete-**Co**ntinuous Latent Variable **Diff**usion Models (DisCo-Diff)*, DMs augmented with additional *discrete* latent variables that encode additional high-level information about the data and can be used by the main DM to simplify its denoising task (Fig. 1). These discrete latents are inferred through an encoder network and learnt end-to-end together with the DM. Thereby, the discrete latents directly learn to encode information that is beneficial for reducing the DM's score matching objective and making the DM's hard task of mapping simple noise to complex data easier. Indeed, in practice, we find that they significantly reduce the curvature of the DM's generative ODE and reduce the DM training loss in particular for large diffusion times, where denoising is most ambiguous and challenging. In contrast to previous work (Bao et al., 2022; Hu et al., 2023; Harvey & Wood, 2023), we do not rely on domain-specific pre-trained encoder networks, making our framework general

[*]Work done during an internship at NVIDIA. [1]NVIDIA [2]MIT. Correspondence to: Yilun Xu <yilunx@nvidia.com>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

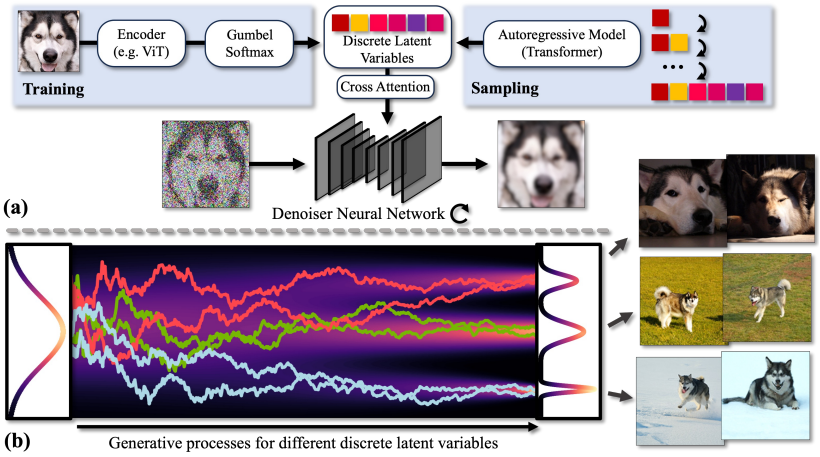


Figure 1. **Dis**crete-**Co**ntinuous Latent Variable **Diff**usion Models (DisCo-Diff) augment DMs with additional *discrete* latent variables that capture global appearance patterns, here shown for images of huskies. **(a)** During training, discrete latents are inferred through an encoder, for images a vision transformer (Dosovitskiy et al., 2021), and fed to the DM via cross-attention. Backpropagation is facilitated by continuous relaxation with a Gumbel-Softmax distribution. To sample novel images, an additional autoregressive model is learnt over the distribution of discrete latents. **(b)** Schematic visualization of generative denoising diffusion trajectories. Different colors indicate different discrete latent variables, pushing the trajectories toward different modes.

and universally applicable. To facilitate sampling of discrete latent variables during inference, we learn an autoregressive model over the discrete latents in a second step. We only use a small set of *discrete* latents with relatively small codebooks, which makes the additional training of the autoregressive model easy. We specifically advocate for the use of auxiliary discrete instead of continuous latents; see Sec. 3.2.

While previous works (Esser et al., 2021; Ramesh et al., 2021; Chang et al., 2022; Yu et al., 2022; Pernias et al., 2023; Chang et al., 2023) use fully discrete latent variable-based approaches to model images, this typically requires large sets of spatially arranged latents with large codebooks, which makes learning their distribution challenging. DisCo-Diff, in contrast, carefully combines its discrete latents with the continuous latents (Gaussian prior) of the DM and effectively separates the modeling of discrete and continuous variations within the data. It requires only a few discrete latents.

To demonstrate its universality, we validate the DisCo-Diff framework on several different tasks. As a motivating example, we study 2D toy distributions, where the discrete latents learn to capture different modes with smaller curvature during sampling. We then tackle image synthesis, where the discrete latents learn large-scale appearance, often associated with global style and color patterns. Thereby, they offer complementary benefits to semantic conditioning information. Quantitatively, DisCo-Diff universally boosts output quality and achieves state-of-the-art performance on several ImageNet generation benchmarks. In addition, we experimentally validate that auxiliary *discrete* latents are superior to *continuous* latents in our setup, and study different network architectures for injecting the discrete latents into the DM network. A careful hierarchical design can encourage different discrete latents to encode different image characteristics, such as shape vs. color, reminiscent of observations from the literature on generative adversarial networks (Karras et al., 2019; 2020). We also apply DisCo-Diff to molecular docking, a critical task in drug discovery, where the discrete latents again improve performance by learning to indicate critical atoms in the interaction and, in this way, deconvolving the multimodal uncertainty given by different possible poses from continuous variability of each pose. Moreover, we augment Poisson Flow Generative Models (Xu et al., 2022; 2023b) with discrete latent variables to showcase that the framework can also be applied to other "iterative" generative models, other than regular DMs, observing similar benefits.

Contributions. *(i)* We propose DisCo-Diff, a novel framework for combining discrete and continuous latent variables in DMs in a universal manner. *(ii)* We extensively validate DisCo-Diff, significantly boosting model quality in all experiments, and achieving state-of-the-art performance on several image synthesis tasks. *(iii)* We present detailed analyses as well as ablation and architecture design studies that demonstrate the unique benefits of discrete latent variables and how they can be fed to the main denoiser network. *(iv)* Overall, we provide insights for designing performant generative models. We make the case for discrete latents by showing that real-world data is best modeled with generative frameworks that leverage *both* discrete and continuous latents. We intentionally developed a simple and universal framework that does not rely on pre-trained encoders to offer a broadly applicable modeling approach to the community.

2. Background

DisCo-Diff builds on (continuous-time) DMs (Song et al., 2021), and we follow the EDM framework (Karras et al., 2022). DMs perturb the clean data $\mathbf{y} \sim p_{\text{data}}(\mathbf{x})$ in a fixed forward process using $\sigma^2(t)$-variance Gaussian noise, where $\mathbf{y} \in \mathbb{R}^d$ and t denotes the time along the diffusion process. The resulting distribution is denoted as $p(\mathbf{x}; \sigma(t))$ with $\mathbf{x} \in \mathbb{R}^d$. For sufficiently large σ_{max}, this distribution is almost identical to pure random Gaussian noise. DMs leverage this observation to sample $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{x}_0; \mathbf{0}, \sigma_{\text{max}}^2 \boldsymbol{I})$ and then iteratively denoise the sample through a sequence of $M+1$ gradually decreasing noise levels $\sigma_{i+1} < \sigma_i$ $(\sigma_0 = \sigma_{\text{max}})$, where $i \in [0, ..., M]$ and $\mathbf{x}_i \sim p(\mathbf{x}; \sigma_i)$. The σ_i correspond to a discretization of a continuous $\sigma(t)$ function. If $\sigma_M = 0$, then the final $\mathbf{x}_M$ follows the data distribution. Sampling

corresponds to simulating a deterministic or stochastic differential equation

$$d\mathbf{x} = \underbrace{-\dot{\sigma}(t)\sigma(t)\nabla_{\mathbf{x}}\log p(\mathbf{x};\sigma(t))dt}_{\text{Probability Flow ODE}} \underbrace{-\beta(t)\sigma^2(t)\nabla_{\mathbf{x}}\log p(\mathbf{x};\sigma(t))dt + \sqrt{2\beta(t)}\sigma(t)d\boldsymbol{\omega}_t}_{\text{Langevin Diffusion SDE}}, \quad (1)$$

where $d\boldsymbol{\omega}_t$ is a standard Wiener process and $\nabla_{\mathbf{x}}\log p(\mathbf{x};\sigma(t))$ is the *score function* of the diffused distribution $p(\mathbf{x};\sigma(t))$. The first term in Eq. (1) is the Probability Flow ODE, which pushes samples from large to small noise levels. The second term is a Langevin Diffusion SDE, an equilibrium sampler for different noise levels $\sigma(t)$, which can help correct errors during synthesis (Karras et al., 2022). This component can be scaled by the time-dependent parameter $\beta(t)$. Setting $\beta(t) = 0$ leads to pure ODE-based synthesis. Generally, different sampling methods can be used to solve the generative ODE/SDE.

Training a DM corresponds to learning a model to approximate the intractable score function $\nabla_{\mathbf{x}}\log p(\mathbf{x};\sigma(t))$. Following the EDM framework, we parametrize $\nabla_{\mathbf{x}}\log p(\mathbf{x};\sigma(t)) = (\boldsymbol{D}_\theta(\mathbf{x},\sigma(t)) - \mathbf{x})/\sigma^2(t)$, where $\boldsymbol{D}_\theta(\mathbf{x},\sigma(t))$ is a learnable *denoiser* neural network that is trained to predict clean data from noisy inputs and is conditioned on the noise level $\sigma(t)$. It can be trained using denoising score matching (Hyvärinen, 2005; Lyu, 2009; Vincent, 2011; Song & Ermon, 2019), minimizing

$$\mathbb{E}_{\mathbf{y}\sim p_{\text{data}}(\mathbf{y})}\mathbb{E}_{t,\mathbf{n}}\left[\lambda(t)||\boldsymbol{D}_\theta(\mathbf{y}+\mathbf{n},\sigma(t)) - \mathbf{y}||^2\right] \quad (2)$$

where $t \sim p(t)$ for a distribution $p(t)$ over diffusion times t, $\mathbf{n} \sim \mathcal{N}(\mathbf{n};\mathbf{0},\sigma^2(t)\boldsymbol{I})$, and $\lambda(t)$ is a function that gives different weight to the objective for different noise levels.

In this work, we use $\sigma(t) = t$ and follow the EDM work's configuration (Karras et al., 2022), unless otherwise noted. Moreover, we also leverage classifier-free guidance in DisCo-Diff when conditioning on the discrete latent variables. Classifier-free guidance combines the score functions of an unconditional and a conditional diffusion model to amplify the conditioning; see Ho & Salimans (2021).

3. DisCo-Diff

In Sec. 3.1, we first formally define DisCo-Diff's generative model and training framework, before discussing and carefully motivating our approach in detail in Sec. 3.2. In Sec. 3.3, we highlight critical architecture considerations.

3.1. Generative Model and Training Objective

In our DisCo-Diff framework (Fig. 1), we augment a DM's learning process with an m-dimensional discrete latent $\mathbf{z} \in \mathbb{N}^m$, where each dimension is a random variable from a categorical distribution of codebook size k. There are three learnable components: the denoiser neural network

(a) 128×128 (b) Shared discrete latents

Figure 2. Samples generated from DisCo-Diff trained on the ImageNet dataset: (a) randomly sampled discrete latents and class labels; (b) samples in each grid sharing the same discrete latent. The class label for the top/bottom row is fixed to coffeepot/malamute.

$\boldsymbol{D}_\theta\colon \mathbb{R}^d \times \mathbb{R} \times \mathbb{N}^m \to \mathbb{R}^d$, corresponding to DisCo-Diff's DM, which predicts denoised images conditioned on diffusion time t and discrete latent $\mathbf{z}$; an encoder $\boldsymbol{E}_\phi\colon \mathbb{R}^d \to \mathbb{N}^m$, used to infer discrete latents given clean images $\mathbf{y}$. It outputs a categorical distribution over the k categories for each discrete latent; and a post-hoc auto-regressive model $\boldsymbol{A}_\psi$, which approximates the distribution of the learned discrete latents $\mathbf{z}$ by $\prod_{i=1}^m p_\psi(\mathbf{z}_i|\mathbf{z}_{<i})$. DisCo-Diff's training process is divided into two stages. In the first stage, the denoiser $\boldsymbol{D}_\theta$ and the encoder $\boldsymbol{E}_\phi$ are co-optimized in an end-to-end fashion. This is achieved by extending the denoising score matching objective (as expressed in Eq. 2) to include learnable discrete latents $\mathbf{z}$ associated with each data $\mathbf{y}$:

$$\mathbb{E}_{\mathbf{y}}\mathbb{E}_{\mathbf{z}\sim\boldsymbol{E}_\phi(\mathbf{y})}\mathbb{E}_{t,\mathbf{n}}\left[\lambda(t)||\boldsymbol{D}_\theta(\mathbf{y}+\mathbf{n},\sigma(t),\mathbf{z}) - \mathbf{y}||^2\right], \quad (3)$$

where $\mathbf{y} \sim p_{\text{data}}(\mathbf{y})$. In contrast to the standard objective in Eq. 2, which focuses on learning the reparameterization of the score $\nabla_{\mathbf{x}}\log p(\mathbf{x};\sigma(t))$, the denoiser in our approach is essentially learning the reparameterization of the conditional score $\nabla_{\mathbf{x}}\log p(\mathbf{x}|\mathbf{z};\sigma(t))$, with the convolution of the probability density functions $p(\cdot|\mathbf{z};\sigma(t)) = p(\cdot|\mathbf{z}) * \mathcal{N}(\mathbf{0},\sigma^2(t)\boldsymbol{I})$. This conditional score originates from conditioning the DM on the discrete latents $\mathbf{z}$, which are inferred by the encoder $\boldsymbol{E}_\phi$. The denoiser network $\boldsymbol{D}_\theta$ can better capture the time-dependent score (*i.e.,* achieving a reduced loss) if the score for each sub-distribution $p(\mathbf{x}|\mathbf{z};\sigma(t))$ is simplified. Therefore, the encoder $\boldsymbol{E}_\phi$, which has access to clean input data, is encouraged to encode useful information into the discrete latents and help the denoiser to more accurately reconstruct the data. Naively backpropagating gradients into the encoder through the sampling of the discrete latent variables $\mathbf{z}$ is not possible. Hence, during training we rely on a continuous relaxation based on the Gumbel-Softmax distribution (Jang et al., 2016) (see App. D for details).

When training the denoiser network, we randomly replace the discrete latent variables with a non-informative null-embedding with probability 0.1. Thereby, the DM learns both a discrete latent variable-conditioned and a regular, unconditional score. During sampling, we can combine

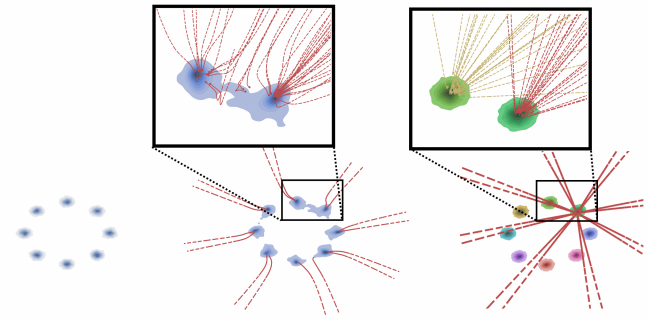


Figure 3. **Modeling 2D mixture of Gaussians.** *Left:* Data distribution. *Middle:* Generated data by regular DM. *Right:* Generated data by DisCo-Diff. We use different colors to distinguish data generated by different discrete latents. We further provide zoom-ins and visualize some ODE trajectories by dotted lines.

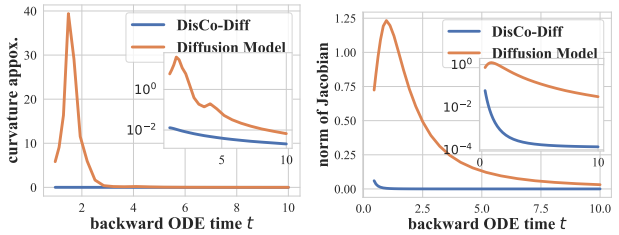


Figure 4. **Modeling 2D mixture of Gaussians: analysis.** The mean curvature *(left)* and norm of the neural networks' Jacobians *(right)* along the reverse-time ODE trajectories as function of t.

these scores for classifier-free guidance (Ho & Salimans, 2021) with respect to the model's own discrete latents, and amplify their conditioning effect (details in App. B).

We can interpret DisCo-Diff as a variational autoencoder (VAE) (Kingma & Welling, 2014; Rezende et al., 2014; van den Oord et al., 2017; Rolfe, 2017) with discrete latents and a DM as decoder. VAEs often employ regularization on their latents. We did not find this to be necessary, as we use only very low-dimensional latent variables, *e.g.,* 10 in our ImageNet experiments, with relatively small codebooks. Moreover, we employ a strictly non-zero temperature in the Gumbel-Softmax relaxation, encouraging stochasticity.

In the second stage, we train the autoregressive model $\boldsymbol{A}_\psi$ to capture the distribution of the discrete latent variables $p_\phi(\mathbf{z})$ defined by pushing the clean data through the trained encoder. We use a maximum likelihood objective as follows:

$$\mathbb{E}_{\mathbf{y}\sim p_{\text{data}}(\mathbf{y}),\mathbf{z}\sim \boldsymbol{E}_\phi(\mathbf{y})}\left[\sum_{i=1}^{m}\log p_\psi(\mathbf{z}_i|\mathbf{z}_{<i})\right] \tag{4}$$

Since we set m to a relatively small number, it becomes very easy for the model to handle such short discrete vectors, which makes this second-stage training efficient. Also the additional sampling overhead due to this autoregressive component on top of the DM becomes negligible. At inference time, when using DisCo-Diff to generate novel samples, we first sample a discrete latent variable from the autoregressive model, and then sample the DM with an ODE or SDE solver. We provide the algorithm pseudocode for training and sampling in Appendix C.

3.2. Motivation and Related Work

We will now critically discuss and motivate our design choices and also discuss the most relevant related works. *For an extended discussion of related work see App. A.*

The curvature of diffusion models. DMs, in their simpler ODE-based formulation ($\beta(t) = 0$ in Eq. (1)), learn a complex noise-to-data mapping. The noise is drawn from an analytically tractable, unimodal Gaussian distribution. As the data is encoded in this distribution, we can consider this high-dimensional Gaussian distribution the DM's *continuous* latent variables (DMs can generally be seen as deep latent variable models (Huang et al., 2021; Kingma et al., 2021)). However, the mapping from unstructured noise to a diverse, typically multimodal data distribution necessarily needs to be highly complex. This corresponds to a highly non-linear generative ODE with strong curvature, which is challenging to learn and also makes synthesis slow by requiring a fine discretization. To illustrate this point, we trained a DM on a simple 2D mixture of Gaussians, where we observe bent ODE trajectories near the data (Fig. 3, *middle*). This effect is significantly stronger in high dimensions.

A simpler mapping with discrete latent variables. The role of the discrete latents in DisCo-Diff is to reduce this complexity and make the DM's learning task easier. The single noise-to-data mapping is effectively partitioned into a set of simpler mappings, each with less curvature in its generative ODE. We argue that it is unnatural to map an entire multimodal complex data distribution to a single continuous Gaussian distribution. Instead, we believe that an ideal generative model should combine both discrete and continuous latent variables, where discrete latents capture global multimodal structure and the continuous latents model local continuous variability. With this in mind, we suggest to only use a moderate number of discrete latents with small codebooks. On the one hand, a few latents can already significantly simplify the DM's learning task. On the other hand, if we only have few latents with small codebooks, training a generative model—an autoregressive one in our case—over the discrete latent variable distribution itself, will be simple (which we observe, see Sec. 4).

Validation in 2D. To validate our reasoning, let us revisit the toy 2D mixture of Gaussians. In Fig. 3, *right*, we show the DisCo-Diff model's synthesized data. The discrete latents learn to capture the different modes, and DisCo-Diff's DM component models the individual modes. The DM's ODE trajectories for different latents are now almost perfectly straight, indicating a simple conditional score function. In Fig. 4, *left*, we quantitatively show strongly reduced curvature along the entire diffusion time t. In Fig. 4, *right*, as a measure of network complexity we also show the norms of the Jacobians of the employed denoiser networks.

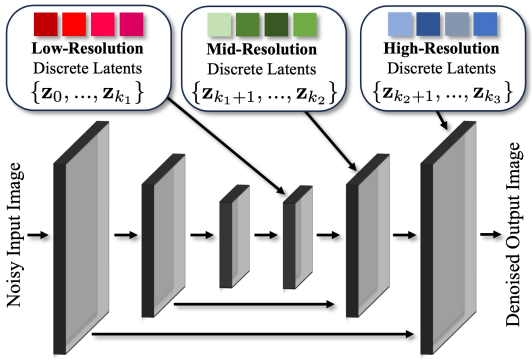


Figure 5. **Group hierarchical DisCo-Diff.** Different discrete latents are fed to the denoiser U-Net at different feature resolutions.

We see significantly reduced norms for DisCo-Diff for all t, suggesting that the denoiser's task is indeed strongly simplified and less network capacity is required.

Using few, global latents with relatively small codebooks is important. DisCo-Diff is fundamentally different from most contemporary generative modeling frameworks using discrete latent variables (van den Oord et al., 2017; Esser et al., 2021; Ramesh et al., 2021; Chang et al., 2022; Yu et al., 2022; Pernias et al., 2023; Chang et al., 2023). These works use autoencoders to encode images in its entirety into spatially-arranged, downsampled representations of the inputs, focusing more on preserving image fidelity than on capturing diverse training data modes. However, this is also unnatural: Encoding continuous variability, like smooth pose, shape, or color variations in images, into discrete latents requires the use of very large codebooks and, on top of that, these models generally rely on very high-dimensional spatial grids of discrete latents (*e.g.* 32x32=1024 latents with codebooks $>1,000$ (Esser et al., 2021), while we use just 10 latents with a codebook size of 100 in our main image models). This makes learning the distribution over the discrete latents very challenging for these types of models, while it is simple in DisCo-Diff, where they just supplement the DM. In DisCo-Diff, we get the best from both continuous and discrete latent variables, using only few *global* latents.

End-to-end training is essential. DisCo-Diff's discrete latents are in spirit similar to leveraging non-learnt conditioning information. As pointed out by Bao et al. (2022), this has been crucial to facilitate training high-performance generative models like strong class-conditional (Dhariwal & Nichol, 2021; Kingma & Gao, 2023) or text-to-image DMs (Ramesh et al., 2022; Ho et al., 2022; Rombach et al., 2022). However, DisCo-Diff aims to fundamentally address the problem, rather than relying on given conditioning data. Moreover, the data usually has significant variability even given, for instance, a class label. Our discrete latents can further reduce the complexity (as observed, see Sec. 4).

However, could we use pre-trained encoder networks, such as CLIP (Radford et al., 2021) or others (He et al., 2020; Caron et al., 2021), to produce encodings to condition on and whose distribution could be modeled in a second stage? This is explored by previous works (Harvey & Wood, 2023; Bao et al., 2022; Hu et al., 2023; Li et al., 2023), but has important disadvantages: *(i)* The most crucial downside is that such encoders are not universally available, but typically only for images. However, we seek to develop a universally applicable framework. For instance, we also apply DisCo-Diff to molecular docking (see Sec. 4.2), where no suitable pre-trained networks are available. *(ii)* In DisCo-Diff, the job of the discrete latents is to make the denoising task of the DM easier, which is especially ambiguous at large noise levels (in fact, we find that the latents help in particular to reduce the loss at these high noise levels, see Fig. 7). It is not obvious what information about the data the latents should best encode for this. By learning them jointly with the DM objective itself, they are directly trained to help the DM learn better denoisers and lower curvature generative ODEs. *(iii)* A generative model needs to be trained over the encodings in the second stage. In DisCo-Diff, we can freely choose an appropriate number of latents and codebook size to simplify the DM's denoising task, while also facilitating easy learning of the autoregressive model in the second stage. When using pre-trained encoders, one must work with the encodings by these methods, which were not developed for generative modeling. We attribute DisCo-Diff's strong generation performance to its end-to-end learning.

The latent variables must be discrete. Could we also use auxiliary continuous latent variables? Generative models on continuous latents are almost always based on mappings from a uni-modal Gaussian distribution to the distribution of latents. Hence, if such continuous latents learnt multimodal structure in the data to simplify the main DM's denoising task, as DisCo-Diff's discrete latents do, then learning a distribution over them in the second stage would again require a highly non-linear difficult-to-learn mapping from Gaussian noise to the multimodal encodings. This is the problem DisCo-Diff aims to solve in the first place. Preechakul et al. (2022) augment DMs with non-spatial continuous latent variables, but they only focus on semantic face image manipulation. InfoDiffusion (Wang et al., 2023) conditions DMs on discrete latent variables. However, it focuses on learning disentangled representations, also primarily for low-resolution face synthesis, and uses a mutual information-based objective. Contrary to DisCo-Diff, neither of these works tackles high-quality synthesis for challenging, diverse datasets.

In our ablation studies (Sec. 4.1), we further validate our design choices and motivations that we presented here.

3.3. Architecture

As discussed, DisCo-Diff enhances the training of continuous DMs by incorporating learnable discrete latent variables that are meant to capture the *global* underlying discrete

Table 1. FID score together with NFE on ImageNet-64.

	FID	NFE
without class-conditioning		
IC-GAN (Casanova et al., 2021)	9.20	1
BigGAN (Brock et al., 2018)	16.90	1
iDDPM (Nichol & Dhariwal, 2021)	16.38	50
EDM (Karras et al., 2022)	6.20	50
SCDM (Bao et al., 2022)	3.94	50
DisCo-Diff (*ours*)	**3.70**	50
class-conditioned, ODE sampler		
EDM (Karras et al., 2022)	2.36	79
PFGM++ (Xu et al., 2023b)	2.32	79
DisCo-PFGM++ (*ours*)	1.92	78
DisCo-Diff (*ours*)	**1.65**	78
class-conditioned, stochastic sampler		
iDDPM (Nichol & Dhariwal, 2021)	2.92	250
ADM (Dhariwal & Nichol, 2021)	2.07	250
CDM (Ho et al., 2021)	1.48	8000
VDM++ (Kingma & Gao, 2023)	1.43	511
EDM (w/ Restart (Xu et al., 2023a))	1.36	623
RIN (Jabri et al., 2022)	1.23	1000
DisCo-Diff (*ours*; w/ Restart (Xu et al., 2023a))	**1.22**	623
class-conditioned, w/ adversarial objective		
IC-GAN (Casanova et al., 2021)	6.70	1
BigGAN-deep (Brock et al., 2018)	4.06	1
CTM (Kim et al., 2023a)	1.92	1
StyleGAN-XL (Sauer et al., 2022)	1.51	1

Table 2. FID score and NFE on class-cond. ImageNet-128.

	FID	NFE
ADM (Dhariwal & Nichol, 2021)	5.91	250
ADM-G (Dhariwal & Nichol, 2021)	2.97	250
CDM (32, 64, 128) (Ho et al., 2021)	3.52	8100
RIN (Jabri et al., 2022)	2.75	1000
VDM++, *w/ ODE sampler* (Kingma & Gao, 2023)	2.29	115
DisCo-Diff, *w/ ODE sampler* (*ours*)	1.98	114
VDM++, *w/ DDPM sampler* (Kingma & Gao, 2023)	1.88	512
DisCo-Diff, *w/ ODE sampler, VDM++ correction*	**1.73**	414

structure of the data. To ensure that DisCo-Diff works as intended, suitable network architectures are necessary. Below, we summarize our design choices, focusing on DisCo-Diff for image synthesis. However, the framework is general, requiring only an encoder to infer discrete latents from clean input data and a conditioning mechanism that integrates these discrete latents into the denoiser network. In fact, we also apply our model to 2D toy data and molecular docking.

Encoder. For image modeling, we utilize a ViT (Dosovitskiy et al., 2021) as the backbone for the encoder. We extend the classification mechanism in ViTs, and treat each discrete token as a different classification token. Concretely, we add m extra classification tokens to the sequence of image patches. This architectural design naturally allows each discrete latent to effectively capture the global characteristic of the images, akin to performing data classification.

Discrete latent variable conditioning. For image experiments, DisCo-Diff's denoisers are U-Nets as widely used for DMs (Karras et al., 2022; Hoogeboom et al., 2023). For the discrete latent variable conditioning, we utilize cross-attention (Rombach et al., 2022). Drawing inspiration from text-to-image generation, DisCo-Diff's discrete latents function analogously to text, exerting a global influence on the denoiser's output. Specifically, image features act as queries and discrete latents are keys and values in the cross-attention layer, enabling discrete latents to globally shape the image features. We add a cross-attention layer after each self-attention layer within the U-Net. In our main models, all discrete latents are given to all cross-attention layers.

Group hierarchical models. To enhance the interpretability of discrete latents, we also explore the inductive bias inherent in the U-Net architecture and feed distinct latent groups into various resolution features in the up-sampling branch of the U-Net, as shown in Fig. 5. This approach draws inspiration from StyleGAN (Karras et al., 2019), where distinct latents are introduced at different resolutions, enabling each to capture different image characteristics by the neural network's inductive bias. This design fosters a group hierarchy, where the groups associated with higher-resolution features offer supplementary information, conditioned upon the groups related to lower-resolution features. We refer to this refined model as the *group hierarchical* DisCo-Diff.

In the **molecular docking** task, existing denoisers operate through message passing in a permutation equivariant way over 3D point clouds representing molecular structures (Corso et al., 2023). We build this property and architectural bias directly into the latent variables, allowing them to take values indicating one node in the point cloud (therefore, for every point cloud, the codebook size equals the number of nodes). This latent design choice aligns with the intuition of the encoder determining the atoms playing key roles in the structure and allows for minimal modification of the score model where the latents simply represent additional features for every node. The encoder is also composed of a similar equivariant message passing, `e3nn` (Geiger & Smidt, 2022), network where for each node one logit per latent will be predicted. More details on the architecture for the molecular docking task can be found in App. E.4.

The **auto-regressive model** over the distribution of the discrete latents is implemented in image experiments using a standard Transformer decoder (Vaswani et al., 2017). For molecular docking, it again uses an `e3nn` network that is fed the conditioning information of the protein structure and molecular graph. Generally, DisCo-Diff is compatible with other conditional inputs, *e.g.* class labels, which can be added as inputs to denoiser and auto-regressive model. We use an auto-regressive model for simplicity and expect DisCo-Diff's second stage to work equally well with other discrete data generative models, *e.g.* discrete state diffusion models (Austin et al., 2021; Campbell et al., 2022). Architecture details, also for 2D toy data experiments, in App. F.

An important question surrounding the architecture design is **how to choose an appropriate number of latents and**

Table 3. Ablations on class-cond. ImageNet-64.

	FID
EDM (Karras et al., 2022)	2.36
Oracle setting	
Continuous latent (KLD weight=0.1)	1.67
Continuous latent (KLD weight=1)	2.36
DisCo-Diff (cfg=0)	1.65
Generative prior on latent	
Continuous latent (KLD weight=0.1)	11.12
Continuous latent (KLD weight=1, cfg=0)	2.36
Continuous latent (KLD weight=1, cfg=1)	2.36
DisCo-Diff (cfg=0)	1.81
DisCo-Diff (cfg=1)	**1.65**
DisCo-Diff (cfg=2)	2.33

codebook size. While intuitively, increasing the number of latents and the codebook size might seem like a straightforward method to reduce the reconstruction error further by capturing more intricate data structures, this approach also introduces additional complexity. Specifically, a larger set of latents or an expanded codebook size complicates the auto-regressive model's task, potentially leading to increased errors. Thus, finding a balance between enhancing model performance through more detailed discrete structures and maintaining manageable modeling complexity for the auto-regressive model is crucial. We recommend using a modest number of latent (e.g. 10 30) and codebook size (e.g. 50 100) for current diffusion models and leaving the exploration of optimal hyper-parameters to future works. For example, in our image generation experiments, we found that a configuration of 10 latents with a codebook size of 100 significantly enhances performance on the complex ImageNet dataset. We did not tune this hyper-parameter due to computational constraints. We believe that more careful hyper-parameter optimization over the exact number of latents and the codebook size would further boost the performance of DisCo-Diff.

4. Experiments

4.1. Image Synthesis

We use the ImageNet (Deng et al., 2009) dataset and tackle both class-conditional (at varying resolutions 64×64 and 128×128) and unconditional synthesis. To measure sample quality, we follow the literature and use Fréchet Inception Distance (FID) (Heusel et al., 2017) (lower is better). We also report the number of neural function evaluations (NFE).

In the class-conditional setting, the DisCo-Diff's denoiser is initialized using pre-trained ImageNet models, except for the new components: the cross-attention layers between discrete latents and images, and the encoder. We fine-tune the pre-trained U-Net in EDM (Karras et al., 2022) with discrete latents for ImageNet-64. For ImageNet-128, we implement the U-ViT in VDM++ (Kingma & Gao, 2023), and fine-tune our trained VDM++ model using discrete latents. We also adhere to their respective noise schedules and loss weightings during the training process. We use Heun's second-order method as ODE sampler, and a 12-layer Transformer as the auto-regressive model. We set the latent dimension to $m = 10$ and the codebook size to $k = 100$ in DisCo-Diff.

Results. See Tables 1 and 2. **(1) DisCo-Diff achieves the new state-of-the-art on class-conditioned ImageNet-64/ImageNet-128 when using ODE sampler.** Specifically, DisCo-Diff reduces the previous state-of-the-art FID score from 2.36 to 1.65 on ImageNet-64, and from 2.29 to 1.98 on ImageNet-128. This aligns with our analysis (Sec. 3.2) that DisCo-Diff yields straighter ODE trajectories.

(2) DisCo-Diff outperforms all baselines in the unconditional setting, or when using stochastic sampler. DisCo-Diff also surpasses the previous best method (SCDM (Bao et al., 2022)) in the unconditional setting, even though their method relies on pre-trained MoCo features. In addition, DisCo-Diff sets the new record ImageNet-64 FID of 1.22 when using Restart sampler (Xu et al., 2023a). Note that the competitive method RIN (Jabri et al., 2022) employs a novel architecture distinct from conventional U-Nets/U-ViTs.

On ImageNet-128, we observe that DisCo-Diff does not perform well with stochastic samplers. When using the DDPM sampler as in VDM++ (Kingma & Gao, 2023), DisCo-Diff achieves an FID score of 2.80, which is worse than VDM++'s FID score of 1.88. As the discrete latents reduce the loss at larger times (*c.f.* Figure 7) and the training targets at these times typically correspond to low-frequency components of images, we hypothesize that in this model the discrete latents learnt to overly emphasize global information, diverting the model to overlook some high-frequency details necessary at smaller times. We provide empirical evidence in Appendix G.3 to show that VDM++ w/ DDPM better captures details at smaller times, which supports this hypothesis. Note that, in theory, VDM++ and DisCo-Diff should perform similarly at smaller times. A potential solution could concatenate the discrete latents with a null token, similar to text-to-image models (Balaji et al., 2022), allowing the model to learn more easily to exclude the influence of discrete latents at smaller times. We leave it for future exploration. We would like to emphasize that we only observed this behavior for this one model and dataset. In all other experiments, discrete latents universally improved performance for all stochastic and non-stochastic samplers, and when used for all times t. To better utilize the DDPM sampler for the current model, we substituted the DisCo-Diff ODE with the VDM++ DDPM trajectories at smaller times ($t < 10$), (DisCo-Diff, *w/ ODE sampler, VDM++ correction* in Table 2), which improves the FID to 1.73. In conclusion, while the discrete latents did not help at small times here, they still boosted performance at larger times and allowed us to outperform the pure VDM++ model and,

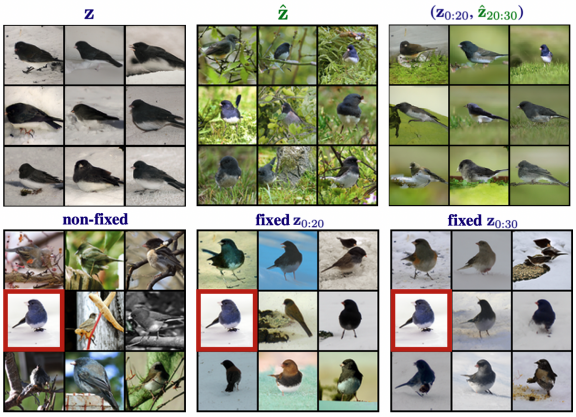


Figure 6. Top: Images created from two 30-dim discrete latents $\mathbf{z}$ and $\hat{\mathbf{z}}$, with the far-right column combining their sub-coordinates. *Bottom:* Variations in images by fixing portions of $\mathbf{z}$ (originating from the red-boxed image). We see that lower-resolution latents affect layout / shape; high-resolution latents alter color / texture.

once again, achieve state-of-the-art performance.

(3) Discrete latents capture variability complementary to class semantics. Fig. 2 (b) illustrates that samples sharing the same discrete latent exhibit similar characteristics, and there are noticeable distinctions for different discrete latents under the same class. It suggests that the discrete latents capture variations that are useful in simplifying the diffusion process defined in Euclidean space beyond class labels, underpinning the improvements of DisCo-Diff over the pre-trained class-conditioned DMs. **(4) Discrete latents boost the performance on PFGM++.** When applied to another ODE-based generative model PFGM++ (Xu et al., 2023b), DisCo-PFGM++ also improves over the baseline version (see Table 1). More samples in App. G.

Ablations and Analyses. Table 3 shows that employing moderate classifier-free guidance with respect to the discrete latents (scale cfg=1) enhances the FID score (studied using ODE sampler), implying that the discrete latents effectively learn modes similar to the role of class labels and text. We further substituted the discrete latents with 1000-dim. continuous latents (1000 to offer capacity at least as high as with the m=10 and k=100 discrete latents), using Kullback-Leibler divergence-based (KLD) regularization as in VAEs to control the information retained. For fair comparison, we trained a DiT-based DM (Peebles & Xie, 2023) on the continuous latents using the same Transformer architecture as in DisCo-Diff's auto-regressive model. Table 3 shows that with a low KLD weight (0.1), the continuous latents are under-regularized, challenging the DiT in modeling the complex encoding distribution and leading to a significant gap between oracle FID (latents predicted from training images) and generative FID (latents sampled from second-stage latent generative models). Conversely, a higher KLD weight (1) causes encoder collapse, and the continuous latents are not used (no latent (EDM), oracle latents and generative latents all produce same FIDs). In contrast, DisCo-Diff's generative FID shows only a minor degradation compared to the oracle FID, indicating the ease of modeling the discrete prior with a simple Transformer.

The DM training objective (Eq. (2)) has most variability at large diffusion times due to the multimodal posterior of clean data given noisy inputs (Xu et al., 2023c). Conditioning information can reduce this ambiguity. For instance, Balaji et al. (2022) show that text conditioning primarily influences the denoiser at larger times. Fig. 7 (a) shows that the learned discrete latents behave similarly to text conditioning, significantly lowering the training loss at higher time steps. Complementarily, Fig. 7 (b) indicates that switching discrete latents towards the end of sampling barely affects the samples, implying they are not used at smaller times t.

In DisCo-Diff, the sampling time of the auto-regressive model is negligible compared to the DM's. For instance, for generating 32 images on ImageNet-128, the auto-regressive models requires only 0.44 seconds, while DisCo-Diff's DM component takes 78 seconds for 114 NFE, with an average of 0.68 second/NFE, all on a single NVIDIA A100 GPU.

Group Hierarchical DisCo-Diff. We evaluate the group hierarchical DisCo-Diff (Sec. 3.3), feeding three separate 10-dim. discrete latents into the U-Net at each level of resolution. Fig. 6 shows that latents for lower-resolution features mainly govern overall shape and layout, while latents for higher-resolution control color and texture. For example, in the bottom figure, when gradually fixing groups in order, the images first converge in shape and then in color.

4.2. Molecular Docking

We test DisCo-Diff also on molecular docking, a fundamental task in drug discovery that consists of generating the 3D structure with which a small molecule will bind to a protein. We build on top of DiffDock (Corso et al., 2023), a DM that recently achieved state-of-the-art performance, integrating discrete latent variables (see Sec. 3 and App. E.4 for details). For computational reasons, we use the reduced DiffDock's architecture (referred to as DiffDock-S) from Corso et al. (2024), which, although less accurate, is much faster for training and inference. For training and evaluation, we follow the standard from Stärk et al. (2022) using the PDBBind dataset (Liu et al., 2017) (see App. E.5).

Results. Table 4 reports performance of our (*DisCo-DiffDock-S*) and relevant baseline methods. We see that also in this domain discrete latents provide improvements, with the success rate on the full dataset increasing from 32.9% to 35.4% and from 13.9% to 18.5% when considering only test complexes with unseen proteins. This improvement is particularly strong on the harder component of the test set, where the baseline model is, likely, highly uncertain. This supports the intuition that DisCo-Diff boosts performance by more appropriately modeling discrete and continuous

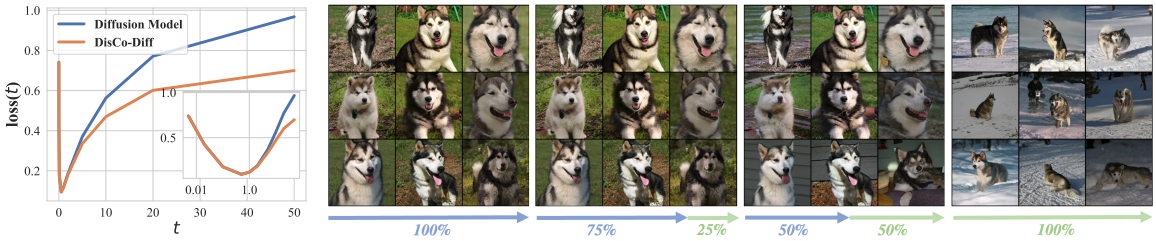


Figure 7. Left: Loss versus time. *Right:* Impact of discrete latent switching during the iterative sampling process of DisCo-Diff's diffusion model component. The numbers represent the percentage of the total sampling steps. The blue/green arrows mean the sampling steps that utilize the discrete latent associated with the leftmost/rightmost grid in the figure. For the middle two images, the process involves initially employing the discrete latent from the leftmost grid for a certain proportion of the total sampling steps (e.g., 75%), before transitioning to the discrete latent from the rightmost grid to complete the remaining steps (e.g., the last 25% of the total sampling steps).

variations in the data. In Fig. 8, we visualize two examples from the test set which highlight how the model learns to associate distinct sets of poses with different latents, decomposing the multimodal components of the pose distribution from the continuous variations that each pose can have.

Table 4. Molecular docking performance on PDBBind. For each method, we report the percentage of top-1 predictions within 2Å of the ground truth for the *full* test set and the subset restricted to *unseen* proteins. Runtime in seconds (* refers to run on CPU).

	Full	Unseen	Runtime
GNINA (McNutt et al., 2021)	22.9	14.0	127
SMINA (Koes et al., 2013)	18.7	14.0	126*
GLIDE (Halgren et al., 2004)	21.8	19.6	1405*
EquiBind (Stärk et al., 2022)	5.5	0.7	0.04
TankBind (Lu et al., 2022)	20.4	6.3	0.7
DiffDock-S (Corso et al., 2024)	32.9	13.9	8.1
DisCo-DiffDock-S (*ours*)	35.4	18.5	9.1
DiffDock (Corso et al., 2023)	38.2	20.8	40

5. Conclusions

We have proposed ***Dis**crete-**Co**ntinuous Latent Variable **Diff**usion Models (DisCo-Diff)*, a novel and universal framework for combining discrete latent variables with continuous DMs. The approach significantly boosts performance by simplifying the DM's denoising task through the help of auxiliary discrete latent variables, while introducing negligible overhead. Extensive experiments and analyses demonstrate the unique benefits of global discrete latent variables that are learnt end-to-end with the denoiser. DisCo-Diff does not rely on any pre-trained encoder networks. As such, we validated our method not only on image synthesis, but also for molecular docking, demonstrating its universality.

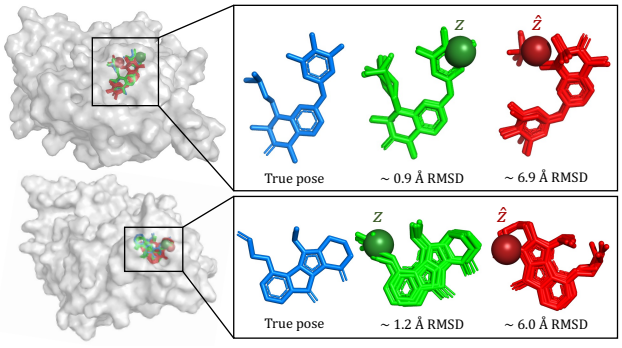


Figure 8. Examples of alternative docking poses modeled when conditioning on different discrete latents, the "correct" z (i.e. same as the encoder) and an incorrect $\hat{z}$. The DM maps them to two distinct sets of plausible orientations with which the ligand could fit in the pocket. Notably, the correct latent corresponds to poses within 2Å of the ground truth. The colored beads are set on the atoms corresponding to the first latent variable.

Limitations and Future Work. There are several potential future directions and limitations in both the experiments and design of DisCo-Diff. First, our experiments have been primarily focused on standard benchmarks such as ImageNet. With more compute resources, DisCo-Diff could be further validated on tasks such as text-to-image generation, where we would expect discrete latent variables to offer complementary benefits to the text conditioning, similar to how discrete latents boost performance in our class-conditional experiments. Moreover, DisCo-Diff could be applied to modalities like speech or 3D data. Secondly, the Group Hierarchical model relies on inductive biases in its architecture, such as the different image characteristics captured at different resolutions in the U-Net. It would be interesting to explore how such architectures could be constructed and similar hierarchical effects could be achieved when working with different data modalities (molecules, etc.). Thirdly, one could apply the idea of DisCo-Diff to other continuous flow models, such as flow-matching (Lipman et al., 2022) or rectified flow (Liu et al., 2022), to further boost their performance. Conceptually, due to the close relation between diffusion models and flow matching, we expect discrete latents to behave similarly there and improve performance. Finally, the current DisCo-Diff framework leverages a two-stage training process. Initially, we jointly train the denoiser and the encoder, followed by the post-hoc auto-regressive model in the second stage. Future work could investigate combining the two-stage training into a seamless end-to-end fashion.

Impact Statement

Deep generative modeling is a burgeoning research field with widespread implications for science and society. Our model DisCo-Diff advances the modeling power of diffusion models for data generation. While enhancing data generation capabilities, notably in high-quality image and video creation, these models also present challenges, such as the potential misuse in deepfake technology leading to social engineering concerns. Addressing these issues necessitates further research into watermark algorithms for diffusion models and collaboration with socio-technical disciplines to balance innovation with ethical considerations. We would also like to highlight the promise of deep generative models like DisCo-Diff in the natural sciences, as exemplified by our molecular docking experiments. Such models have the potential to provide novel insights into, for instance, the interactions between proteins and ligands and advance drug discovery.

Acknowledgements

YX and TJ acknowledge support from MIT-DSTA Singapore collaboration; YX, GC, and TJ from NSF Expeditions grant (award 1918839) "Understanding the World Through Code", from MIT-IBM Grand Challenge project, NSF Expeditions grant (award 1918839: Collaborative Research: Understanding the World Through Code), Machine Learning for Pharmaceutical Discovery and Synthesis (MLPDS) consortium; GC and TJ further acknowledge support from the Abdul Latif Jameel Clinic for Machine Learning in Health and the DTRA Discovery of Medical Countermeasures Against New and Emerging (DOMANE) threats program.

References

Austin, J., Johnson, D. D., Ho, J., Tarlow, D., and van den Berg, R. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, 2021.

Bahmani, S., Skorokhodov, I., Rong, V., Wetzstein, G., Guibas, L., Wonka, P., Tulyakov, S., Park, J. J., Tagliasacchi, A., and Lindell, D. B. 4d-fy: Text-to-4d generation using hybrid score distillation sampling. *arXiv preprint arXiv:2311.17984*, 2023.

Balaji, Y., Nah, S., Huang, X., Vahdat, A., Song, J., Zhang, Q., Kreis, K., Aittala, M., Aila, T., Laine, S., Catanzaro, B., Karras, T., and Liu, M.-Y. eDiff-I: Text-to-Image Diffusion Models with Ensemble of Expert Denoisers. *arXiv preprint arXiv:2211.01324*, 2022.

Bao, F., Li, C., Sun, J., and Zhu, J. Why are conditional generative models better than unconditional ones? *arXiv preprint arXiv:2212.00362*, 2022.

Blattmann, A., Rombach, R., Ling, H., Dockhorn, T., Kim, S. W., Fidler, S., and Kreis, K. Align your Latents: High-Resolution Video Synthesis with Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Brock, A., Donahue, J., and Simonyan, K. Large scale gan training for high fidelity natural image synthesis. *ArXiv*, abs/1809.11096, 2018.

Campbell, A., Benton, J., Bortoli, V. D., Rainforth, T., Deligiannidis, G., and Doucet, A. A continuous time framework for discrete denoising models. In *Advances in Neural Information Processing Systems*, 2022.

Caron, M., Touvron, H., Misra, I., Jegou, H., Mairal, J., Bojanowski, P., and Joulin, A. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Casanova, A., Careil, M., Verbeek, J., Drozdzal, M., and Romero-Soriano, A. Instance-conditioned gan. In *Neural Information Processing Systems*, 2021.

Chang, H., Zhang, H., Jiang, L., Liu, C., and Freeman, W. T. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Chang, H., Zhang, H., Barber, J., Maschinot, A., Lezama, J., Jiang, L., Yang, M.-H., Murphy, K. P., Freeman, W. T., Rubinstein, M., Li, Y., and Krishnan, D. Muse: Text-to-image generation via masked generative transformers. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023.

Corso, G., Stärk, H., Jing, B., Barzilay, R., and Jaakkola, T. Diffdock: Diffusion steps, twists, and turns for molecular docking. *International Conference on Learning Representations (ICLR)*, 2023.

Corso, G., Deng, A., Polizzi, N., Barzilay, R., and Jaakkola, T. The discovery of binding modes requires rethinking docking generalization. In *International Conference on Learning Representations*, 2024.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. ImageNet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition (CVPR)*, 2009.

Dhariwal, P. and Nichol, A. Q. Diffusion Models Beat GANs on Image Synthesis. In *Advances in Neural Information Processing Systems*, 2021.

Dockhorn, T., Vahdat, A., and Kreis, K. Genie: Higher-order denoising diffusion solvers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022a.

Dockhorn, T., Vahdat, A., and Kreis, K. Score-based generative modeling with critically-damped langevin diffusion. In *International Conference on Learning Representations (ICLR)*, 2022b.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Esser, P., Rombach, R., and Ommer, B. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Ge, S., Nah, S., Liu, G., Poon, T., Tao, A., Catanzaro, B., Jacobs, D., Huang, J.-B., Liu, M.-Y., and Balaji, Y. Preserve Your Own Correlation: A Noise Prior for Video Diffusion Models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Geiger, M. and Smidt, T. e3nn: Euclidean neural networks. *arXiv preprint arXiv:2207.09453*, 2022.

Guo, H., Liu, S., Mingdi, H., Lou, Y., and Jing, B. Diffdocksite: A novel paradigm for enhanced protein-ligand predictions through binding site identification. In *NeurIPS 2023 Generative AI and Biology (GenBio) Workshop*, 2023.

Halgren, T. A., Murphy, R. B., Friesner, R. A., Beard, H. S., Frye, L. L., Pollard, W. T., and Banks, J. L. Glide: a new approach for rapid, accurate docking and scoring. 2. enrichment factors in database screening. *Journal of medicinal chemistry*, 2004.

Harvey, W. and Wood, F. Visual chain-of-thought diffusion models. *arXiv preprint arXiv:2303.16187*, 2023.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems*, 2017.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017.

Hinton, G. E. A practical guide to training restricted boltzmann machines. In *Neural networks: Tricks of the trade*, pp. 599–619. Springer, 2012.

Ho, J. and Salimans, T. Classifier-Free Diffusion Guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

Ho, J., Jain, A., and Abbeel, P. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Ho, J., Saharia, C., Chan, W., Fleet, D. J., Norouzi, M., and Salimans, T. Cascaded diffusion models for high fidelity image generation. *J. Mach. Learn. Res.*, 23:47:1–47:33, 2021.

Ho, J., Chan, W., Saharia, C., Whang, J., Gao, R., Gritsenko, A., Kingma, D. P., Poole, B., Norouzi, M., Fleet, D. J., and Salimans, T. Imagen Video: High Definition Video Generation with Diffusion Models. *arXiv preprint arXiv:2210.02303*, 2022.

Hoogeboom, E., Heek, J., and Salimans, T. Simple Diffusion: End-to-End Diffusion for High Resolution Images. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023.

Hu, V. T., Zhang, D. W., Asano, Y. M., Burghouts, G. J., and Snoek, C. G. M. Self-guided diffusion models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Huang, C.-W., Lim, J. H., and Courville, A. A variational perspective on diffusion-based generative models and score matching. In *Neural Information Processing Systems (NeurIPS)*, 2021.

Hyvärinen, A. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6:695–709, 2005. ISSN 1532-4435.

Ingraham, J., Baranov, M., Costello, Z., Frappier, V., Ismail, A., Tie, S., Wang, W., Xue, V., Obermeyer, F., Beam, A., and Grigoryan, G. Illuminating protein space with a programmable generative model. *Nature*, 623:1070–1078, 2023.

Jabri, A., Fleet, D. J., and Chen, T. Scalable adaptive computation for iterative generation. In *International Conference on Machine Learning*, 2022.

Jang, E., Gu, S. S., and Poole, B. Categorical reparameterization with gumbel-softmax. *ArXiv*, abs/1611.01144, 2016.

Karras, T., Laine, S., and Aila, T. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., and Aila, T. Analyzing and improving the image

quality of StyleGAN. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Karras, T., Aittala, M., Aila, T., and Laine, S. Elucidating the design space of diffusion-based generative models. *ArXiv*, abs/2206.00364, 2022.

Ketata, M. A., Laue, C., Mammadov, R., Stärk, H., Wu, M., Corso, G., Marquet, C., Barzilay, R., and Jaakkola, T. S. Diffdock-pp: Rigid protein-protein docking with diffusion models. *arXiv preprint arXiv:2304.03889*, 2023.

Kim, D., Lai, C.-H., Liao, W.-H., Murata, N., Takida, Y., Uesaka, T., He, Y., Mitsufuji, Y., and Ermon, S. Consistency trajectory models: Learning probability flow ode trajectory of diffusion. *ArXiv*, abs/2310.02279, 2023a.

Kim, S. W., Brown, B., Yin, K., Kreis, K., Schwarz, K., Li, D., Rombach, R., Torralba, A., and Fidler, S. NeuralField-LDM: Scene Generation with Hierarchical Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023b.

Kingma, D. P. and Gao, R. Understanding diffusion objectives as the ELBO with simple data augmentation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. In *The International Conference on Learning Representations*, 2014.

Kingma, D. P., Salimans, T., Poole, B., and Ho, J. Variational diffusion models. In *Advances in Neural Information Processing Systems*, 2021.

Koes, D. R., Baumgartner, M. P., and Camacho, C. J. Lessons learned in empirical scoring with smina from the csar 2011 benchmarking exercise. *Journal of chemical information and modeling*, 2013.

Krivák, R. and Hoksza, D. P2rank: machine learning based tool for rapid and accurate prediction of ligand binding sites from protein structure. *Journal of cheminformatics*, 10:1–12, 2018.

Landrum, G. Rdkit documentation. *Release*, 2013.

Li, T., Katabi, D., and He, K. Self-conditioned image generation via generating representations. *arXiv preprint arXiv:2312.03701*, 2023.

Ling, H., Kim, S. W., Torralba, A., Fidler, S., and Kreis, K. Align your gaussians: Text-to-4d with dynamic 3d gaussians and composed diffusion models. *arXiv preprint arXiv:2312.13763*, 2023.

Lipman, Y., Chen, R. T. Q., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling. *ArXiv*, abs/2210.02747, 2022. URL https://api.semanticscholar.org/CorpusID:252734897.

Liu, R., Wu, R., Van Hoorick, B., Tokmakov, P., Zakharov, S., and Vondrick, C. Zero-1-to-3: Zero-shot One Image to 3D Object. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Liu, X., Gong, C., and Liu, Q. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

Liu, Z., Su, M., Han, L., Liu, J., Yang, Q., Li, Y., and Wang, R. Forging the basis for developing protein–ligand interaction scoring functions. *Accounts of Chemical Research*, 2017.

Lu, W., Wu, Q., Zhang, J., Rao, J., Li, C., and Zheng, S. Tankbind: Trigonometry-aware neural networks for drug-protein binding structure prediction. *Advances in neural information processing systems*, 2022.

Lyu, S. Interpretation and generalization of score matching. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, pp. 359–366, Arlington, Virginia, USA, 2009. AUAI Press.

Masters, M. R., Mahmoud, A. H., and Lill, M. A. Fusiondock: Physics-informed diffusion model for molecular docking. *ICML Workshop on Computational Biology*, 2023.

McNutt, A. T., Francoeur, P., Aggarwal, R., Masuda, T., Meli, R., Ragoza, M., Sunseri, J., and Koes, D. R. Gnina 1.0: molecular docking with deep learning. *Journal of cheminformatics*, 2021.

Nichol, A. and Dhariwal, P. Improved denoising diffusion probabilistic models. *ArXiv*, abs/2102.09672, 2021.

Nichol, A., Jun, H., Dhariwal, P., Mishkin, P., and Chen, M. Point-E: A System for Generating 3D Point Clouds from Complex Prompts, 2022.

Peebles, W. and Xie, S. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.

Pernias, P., Rampas, D., Richter, M. L., Pal, C. J., and Aubreville, M. Wuerstchen: An efficient architecture for large-scale text-to-image diffusion models. *arXiv preprint arXiv:2306.00637*, 2023.

Plainer, M., Toth, M., Dobers, S., Stärk, H., Corso, G., Marquet, C., and Barzilay, R. Diffdock-pocket: Diffusion

for pocket-level docking with sidechain flexibility. In *NeurIPS 2023 Workshop on New Frontiers of AI for Drug Discovery and Development*, 2023.

Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., and Rombach, R. SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

Poole, B., Jain, A., Barron, J. T., and Mildenhall, B. DreamFusion: Text-to-3D using 2D Diffusion. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023.

Preechakul, K., Chatthee, N., Wizadwongsa, S., and Suwajanakorn, S. Diffusion autoencoders: Toward a meaningful and decodable representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning Transferable Visual Models From Natural Language Supervision. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021.

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. Zero-shot text-to-image generation. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical Text-Conditional Image Generation with CLIP Latents. *arXiv preprint arXiv:2204.06125*, 2022.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, 2014.

Rolfe, J. T. Discrete variational autoencoders. In *International Conference on Learning Representations*, 2017.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-Resolution Image Synthesis with Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S. K. S., Gontijo-Lopes, R., Ayan, B. K., Salimans, T., Ho, J., Fleet, D. J., and Norouzi, M. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Salakhutdinov, R. and Hinton, G. Deep boltzmann machines. In *Artificial intelligence and statistics*, 2009.

Sauer, A., Schwarz, K., and Geiger, A. Stylegan-xl: Scaling stylegan to large diverse datasets. *ACM SIGGRAPH 2022 Conference Proceedings*, 2022.

Schütt, K., Kindermans, P.-J., Sauceda Felix, H. E., Chmiela, S., Tkatchenko, A., and Müller, K.-R. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. *Advances in neural information processing systems*, 30, 2017.

Schwarz, K., Kim, S. W., Gao, J., Fidler, S., Geiger, A., and Kreis, K. WildFusion: Learning 3D-Aware Latent Diffusion Models in View Space. *arXiv preprint arXiv:2311.13570*, 2023.

Singer, U., Polyak, A., Hayes, T., Yin, X., An, J., Zhang, S., Hu, Q., Yang, H., Ashual, O., Gafni, O., Parikh, D., Gupta, S., and Taigman, Y. Make-A-Video: Text-to-Video Generation without Text-Video Data. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023a.

Singer, U., Sheynin, S., Polyak, A., Ashual, O., Makarov, I., Kokkinos, F., Goyal, N., Vedaldi, A., Parikh, D., Johnson, J., and Taigman, Y. Text-to-4D Dynamic Scene Generation. In *Proceedings of the 40th International Conference on Machine Learning*, 2023b.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. In *International Conference on Machine Learning (ICML)*, 2015.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. In *Proceedings of the 33rd Annual Conference on Neural Information Processing Systems*, 2019.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-Based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations (ICLR)*, 2021.

Stärk, H., Ganea, O., Pattanaik, L., Barzilay, R., and Jaakkola, T. Equibind: Geometric deep learning for drug binding structure prediction. In *International Conference on Machine Learning*, 2022.

Trott, O. and Olson, A. J. Autodock vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of computational chemistry*, 2010.

Vahdat, A., Andriyash, E., and Macready, W. G. Dvae#: Discrete variational autoencoders with relaxed Boltzmann priors. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018a.

Vahdat, A., Macready, W., Bian, Z., Khoshaman, A., and Andriyash, E. DVAE++: Discrete variational autoencoders with overlapping transformations. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018b.

Vahdat, A., Kreis, K., and Kautz, J. Score-based Generative Modeling in Latent Space. In *Neural Information Processing Systems (NeurIPS)*, 2021.

van den Oord, A., Vinyals, O., and kavukcuoglu, k. Neural discrete representation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Vaswani, A., Shazeer, N. M., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In *Neural Information Processing Systems*, 2017.

Vincent, P. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011.

Wang, Y., Schiff, Y., Gokaslan, A., Pan, W., Wang, F., De Sa, C., and Kuleshov, V. InfoDiffusion: Representation learning using information maximizing diffusion models. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023.

Watson, J. L., Juergens, D., Bennett, N. R., Trippe, B. L., Yim, J., Eisenach, H. E., Ahern, W., Borst, A. J., Ragotte, R. J., Milles, L. F., Wicky, B. I. M., Hanikel, N., Pellock, S. J., Courbet, A., Sheffler, W., Wang, J., Venkatesh, P., Sappington, I., Torres, S. V., Lauko, A., Bortoli, V. D., Mathieu, E., Barzilay, R., Jaakkola, T. S., DiMaio, F., Baek, M., and Baker, D. De novo design of protein structure and function with rfdiffusion. *Nature*, 620: 1089–1100, 2023.

Xu, Y., Liu, Z., Tegmark, M., and Jaakkola, T. Poisson flow generative models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Xu, Y., Deng, M., Cheng, X., Tian, Y., Liu, Z., and Jaakkola, T. Restart sampling for improving generative processes. *ArXiv*, abs/2306.14878, 2023a.

Xu, Y., Liu, Z., Tian, Y., Tong, S., Tegmark, M., and Jaakkola, T. PFGM++: Unlocking the potential of physics-inspired generative models. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023b.

Xu, Y., Tong, S., and Jaakkola, T. Stable target field for reduced variance score estimation in diffusion models. *ArXiv*, abs/2302.00670, 2023c.

Yim, J., Trippe, B. L., Bortoli, V. D., Mathieu, E., Doucet, A., Barzilay, R., and Jaakkola, T. Se(3) diffusion model with application to protein backbone generation. *arXiv preprint arXiv:2302.02277*, 2023.

Yu, J., Xu, Y., Koh, J. Y., Luong, T., Baid, G., Wang, Z., Vasudevan, V., Ku, A., Yang, Y., Ayan, B. K., Hutchinson, B., Han, W., Parekh, Z., Li, X., Zhang, H., Baldridge, J., and Wu, Y. Scaling autoregressive models for content-rich text-to-image generation. *Transactions on Machine Learning Research (TMLR)*, 2022.

Zeng, X., Vahdat, A., Williams, F., Gojcic, Z., Litany, O., Fidler, S., and Kreis, K. LION: Latent Point Diffusion Models for 3D Shape Generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Zheng, Y., Li, X., Nagano, K., Liu, S., Kreis, K., Hilliges, O., and Mello, S. D. A unified approach for text- and image-guided 4d scene generation. *arXiv preprint arXiv:2311.16854*, 2023.

Appendix

A. Related Work

Our work builds on DMs (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021; Karras et al., 2022), which have been widely used not only for image generation (Dhariwal & Nichol, 2021; Nichol & Dhariwal, 2021; Rombach et al., 2022; Dockhorn et al., 2022b;a; Saharia et al., 2022; Ramesh et al., 2022; Podell et al., 2023), but also for video (Blattmann et al., 2023; Singer et al., 2023a; Ho et al., 2022; Ge et al., 2023), 3D (Nichol et al., 2022; Zeng et al., 2022; Kim et al., 2023b; Poole et al., 2023; Schwarz et al., 2023; Liu et al., 2023) and 4D (Singer et al., 2023b; Ling et al., 2023; Bahmani et al., 2023; Zheng et al., 2023) synthesis, as well as in various other domains, including, for instance, molecular docking and protein design (Corso et al., 2023; Yim et al., 2023; Ingraham et al., 2023; Watson et al., 2023).

In the DM literature, latent variables have been most popular as part of latent diffusion models, where a DM is trained in a compressed, usually continuous, latent space (Rombach et al., 2022; Vahdat et al., 2021). In contrast, DisCo-Diff leverages *discrete* latent variables and uses them to augment a DM. The first models using discrete latent variables for high-dimensional generative modeling tasks include Boltzmann machines (Salakhutdinov & Hinton, 2009; Hinton, 2012) and early discrete variational autoencoders (Rolfe, 2017; Vahdat et al., 2018a;b). More recently, a variety of works encode images into large 2D spatial grids of discrete tokens with vector quantization or similar techniques (van den Oord et al.,

2017; Esser et al., 2021; Ramesh et al., 2021; Chang et al., 2022; Yu et al., 2022; Pernias et al., 2023; Chang et al., 2023). As discussed, these models typically require a very large number of tokens and rely on large codebooks, which makes modeling their distribution challenging. DisCo-Diff, in contrast, leverages only few discrete latents with small codebooks that act in harmony with the additional continuous DM.

There are previous related works that also condition DMs on auxiliary encodings. Preechakul et al. (2022) augment DMs with non-spatial latent variables, but their latents are continuous and high-dimensional, which makes training their latent DM more challenging. This is precisely what we avoid by instead using low-dimensional and discrete latents. Moreover, they focus on semantic face image manipulation, not high-quality synthesis for challenging, diverse datasets.

Harvey & Wood (2023) use the representations of a pre-trained CLIP image encoder (Radford et al., 2021) for conditioning a DM and learn another DM over the CLIP embeddings for sampling. Similarly, Bao et al. (2022) and Hu et al. (2023) use clustered MoCo-based (He et al., 2020) and clustered DINO-based (Caron et al., 2021) features, respectively, for conditioning. Hence, these three approaches are strictly limited to image synthesis, where such encoders, pre-trained on large-scale datasets, are available. In contrast, we purposefully avoid the use of pre-trained networks and learn the discrete latents jointly with the DM, making our framework universally applicable. Another related work is InfoDiffusion (Wang et al., 2023), which also conditions DMs on discrete latent variables. However, contrary to DisCo-Diff, this work focuses on learning disentangled representations, similar to β-VAEs (Higgins et al., 2017), primarily for low-resolution face synthesis. It uses a mutual information-based objective and does not focus on diverse and high-quality synthesis of complex data such as ImageNet.

In contrast to the above works, we show how discrete latent variables boost generative performance itself and we significantly outperform these works in complex and diverse high-quality synthesis. Furthermore, we motivate DisCo-Diff fundamentally, with reduced ODE curvature and model complexity, providing a new and complementary perspective.

In the molecular docking literature, since DiffDock (Corso et al., 2023) introduced the use of diffusion models in the task, a number of works have proposed different modifications to its framework. In particular, some (Masters et al., 2023; Plainer et al., 2023; Guo et al., 2023) have proposed to separate the blind docking task between pocket identification (i.e. identifying the region of the protein where the small molecule would bind) and pose prediction (i.e. predicting the specific pose with which the ligand would bind to the protein), as previously done in many traditional approaches (Krivák & Hoksza, 2018). One could see this as hand-crafting a (roughly discrete) latent variable in the pocket and using it to decompose the task. By allowing the encoder to learn arbitrary discrete latents through its interaction with the denoiser, DisCo-Diff largely includes the above-mentioned strategy as a particular case.

B. Discrete Latent Variable Classifier-Free Guidance

Classifier-free guidance (Ho & Salimans, 2021) (cfg) is a mode-seeking technique commonly used in diffusion literature, such as class-conditioned genreation (Peebles & Xie, 2023) or text-to-image generation (Rombach et al., 2022). It generally guides the sampling trajectories toward higher-density regions. We can similarly apply classifier-free guidance in the DisCo-Diff, where we treat the discrete latent as conditional inputs. We follow the convention in (Saharia et al., 2022), and the classifier-free guidance at time step t is as follows: $\tilde{\boldsymbol{D}}_\theta(\mathbf{x}, \sigma(t), \mathbf{z}) = w\boldsymbol{D}_\theta(\mathbf{x}, \sigma(t), \mathbf{z}) + (1-w)\boldsymbol{D}_\theta(\mathbf{x}, \sigma(t), \emptyset)$, where $\boldsymbol{D}_\theta(\mathbf{x}, \sigma(t), \mathbf{z})/\boldsymbol{D}_\theta(\mathbf{x}, \sigma(t), \emptyset)$ is the conditional/unconditional models, sharing parameters. We drop the discrete latent with probability 0.1 during training, to train the unconditional model $\boldsymbol{D}_\theta(\mathbf{x}, \sigma(t), \emptyset)$. A mild w would usually lead to improvement in sample diversity (Peebles & Xie, 2023). Table 3 demonstrates that using a moderate guidance scale w=1 (we use $w = 1$ and cfg=1 interchangeably in the paper) improves the FID score, suggesting that the learned discrete latent in the DisCo-Diff framework has strong indications of mode of data distribution. We further explore varying the guidance scale on ImageNet-128. As shown in Fig 9, increasing the classifier-free guidance scale w would strengthen the effect of guidance.

(a) cfg=0 (b) cfg=1 (c) cfg=4 (d) cfg=8

Figure 9. Generated samples in DisCo-Diff with a cfg scale ranging from 0 to 8, under the class label "malamute" on ImageNet-128.

C. Algorithm Pseudocode

We provide algorithm pseudocode for the training in the first stage for denoiser and encoder (Alg 1) and the second stage for auto-regressive model (Alg 2) for clarity. We also include the pseudocode for sampling in Alg 3. Note that we generalize the equations in the main text by considering the conditional generation with condition c.

Algorithm 1 Mini-batch training of denoiser and encoder in DisCo-Diff

Input: Denoiser $\boldsymbol{D}_\theta$, encoder $\boldsymbol{E}_\phi$, training dataset D, batch size $\mathcal{B}$, Gumbel-Softmax temperature τ, training iteration T
for $i = 0, \dots, T-1$ **do**
  Sample mini-batch data $\{(\mathbf{y}_i, c_i)\}_{i=1}^{\mathcal{B}}$ from D
  Sample time variables $\{t_i\}_{i=1}^{\mathcal{B}}$ from $p(t)$ and noise vectors $\{\mathbf{n}_i \sim \mathcal{N}(0, \sigma_i^2 \boldsymbol{I})\}_{i=1}^{\mathcal{B}}$
  Get perturbed data $\{\hat{\mathbf{y}}_i = \mathbf{y}_i + \mathbf{n}_i\}_{i=1}^{\mathcal{B}}$
  Sample the discrete latent from the encoder $\{\mathbf{z}_i \sim \boldsymbol{E}_\phi(\mathbf{y}_i)\}_{i=1}^{\mathcal{B}}$ using Gumbel-Softmax relaxation with temperature τ
  Calculate loss $\ell(\theta, \phi) = \sum_{i=1}^{\mathcal{B}} \lambda(t) \|\boldsymbol{D}_\theta(\hat{\mathbf{y}}_i, t_i, \mathbf{z}_i, c_i) - \mathbf{y}_i\|_2^2$
  Update the network parameter θ and ϕ via Adam optimizer
end for

Algorithm 2 Mini-batch training of auto-regressive model in DisCo-Diff

Input: Auto-regressive model $\boldsymbol{A}_\psi$, encoder $\boldsymbol{E}_\phi$, training dataset D, batch size $\mathcal{B}$, Gumbel-Softmax temperature τ, training iteration T
for $i = 0, \dots, T-1$ **do**
  Sample mini-batch data $\{(\mathbf{y}_i, c_i)\}_{i=1}^{\mathcal{B}}$ from D
  Sample the discrete latent from the encoder $\{\mathbf{z}_i \sim \boldsymbol{E}_\phi(\mathbf{y}_i)\}_{i=1}^{\mathcal{B}}$ using Gumbel-Softmax relaxation with temperature τ
  Calculate loss $\ell(\psi) = \sum_{i=1}^{\mathcal{B}} \sum_j^m \log p_\psi((\mathbf{z}_i)_j | (\mathbf{z}_i)_{<j}, c_i)$
  Update the network parameter ψ via Adam optimizer
end for

D. ImageNet Experiments

D.1. Architecture

	Feature maps	Attention resolution	Encoder patch size
ImageNet-64	1-2-3-4 ($\times$192)	(8,16,32)	8×8
ImageNet-128	1-2-4-16 ($\times$128)	(16,32,64)	16×16

Table 5. Specific network configurations on ImageNet

For all the ImageNet experiments, we fix the latent dimension $m = 10$ in DisCo-Diff, and the codebook size for each

Algorithm 3 Sampling procedure of DisCo-Diff

Input: Auto-regressive model $\boldsymbol{A}_\psi$, denoiser $\boldsymbol{D}_\phi$, time discretization $\{t_i\}_{i=0}^n$, condition c
`/* Sample discrete latent from auto-regressive model */`
for $i = 1, \dots, m$ **do**
  Sample $\mathbf{z}_i \sim p_\psi(\mathbf{z}_i|\mathbf{z}_{<i}, c)$
end for
`/* Diffusion ODE (Heun's 2nd order method) */`
Sample $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, t_0^2\boldsymbol{I})$
for $i = 0, \dots, n-1$ **do**
  $\mathbf{d}_i = (\mathbf{x}_i - \boldsymbol{D}_\theta(\mathbf{x}_i, t_i, \mathbf{z}, c))/t_i$
  $\mathbf{x}_{i+1} = \mathbf{x}_i + (t_{i+1} - t_i)\mathbf{d}_i$
  if $t_{i+1} \neq 0$ **then**
    $\mathbf{d}'_i = (\mathbf{x}_{i+1} - \boldsymbol{D}_\theta(\mathbf{x}_{i+1}, t_{i+1}, \mathbf{z}, c))/t_{i+1}$
    $\mathbf{x}_{i+1} = \mathbf{x}_i + (t_{i+1} - t_i)(\frac{1}{2}\mathbf{d}_i + \frac{1}{2}\mathbf{d}'_i)$
  end if
end for

discrete latent to 1000. Below we provide architecture details for the denoiser network, encoder, and auto-regressive model. Table 5 also lists some key network configurations. Please see the source code in the Supplementary Material for all low-level details.

Denoiser Neural Network. (1) **ImageNet-64**: We use the same UNet architecture in EDM (Karras et al., 2022) for ImageNet-64, with newly injected cross-attention layers after each self-attention layers in each residual block. We feed the discrete vector into a six-layer Transformer encoder (with latent dimension 192) to obtain the corresponding embeddings for discrete variables. These embeddings are then input into the cross-attention layers. (2) **ImageNet-128**: We employ the UViT design in simple diffusion (Hoogeboom et al., 2023) and VDM++ (Kingma & Gao, 2023). UViT uses convolutional layers for down-/up-sampling, and a 36-layer ViT to process the lowest-resolution feature maps in the bottleneck, to strike a better balance between expressiveness and computation. Since the authors didn't release the code and model, we reimplemented the architecture by ourselves. We empirically observe that the convolutional blocks in EDM work better than the ones described in the simple diffusion paper, so we combine the up-/down-sampling blocks in EDM with the 36-layer ViT at the bottleneck layer. We further introduce a cross-attention layer for discrete latent in each up-/down-sampling block, and every three Transformer blocks in the ViT (*e.g.,* 12 new cross-attention layers in the ViT) to save computation.

Encoder. We utilize a 12-layer standard ViT (Dosovitskiy et al., 2021) as the backbone for encoder. Its latent dimension is 768 and the number of attention heads is 12. The patch size for ImageNet-64 is 8×8 and for ImageNet-128 is 16×16. We treat each of the m discrete latents as a classification token, and concatenate their embeddings with the path embeddings.

Auto-regressive model. We use a standard Transformer decoder (Vaswani et al., 2017), with a depth of 12, a number of heads of 8, and a latent dimension of 512. The inference time of the auto-regressive model is much smaller than the iterative denoising process, given that the discrete latent only has 10 dimensions. To generate 32 images on ImageNet-128, the auto-regressive model takes 0.44 seconds, while the diffusion model takes 78 seconds for 114 NFE, with an average of 0.68 s/NFE on a single NVIDIA A100 GPU.

D.2. Training and Sampling

We borrow the preconditioning techniques, training noisy schedule, optimizers, exponential moving average (EMA) schedule, and hyper-parameters from previous state-of-the-art diffusion model EDM (Karras et al., 2022) on ImageNet-64. We employ the shifted EDM-monotonic noisy schedule proposed in VDM++ (Kingma & Gao, 2023) on ImageNet-128, and keep other training details the same in ImageNet-64. We use the Gumbel-Softmax (Jang et al., 2016) as the continuous relaxation for the discrete latents. The temperature τ in Gumbel-Softmax controls the smoothness of the categorical distribution. When $\tau \to 0$, the expected value of the Gumbel-Softmax is the same as the one of the underlying predicted distribution. As we increase t, the Gumbel-Softmax would gradually converge to a uniform distribution. Hence, a relatively large τ effectively provides regualization effects. During training, we set the τ to a constant 1. However, for the extraction of latents from training images, which aids in constructing the dataset for the second stage of the auto-regressive model, we adjust τ to a

lower value of 0.01.

We use Heun's second-order ODE solver as the default ODE sampler, which is proven effective in previous works (Karras et al., 2022; Xu et al., 2023b). We directly use the hyper-parameters for the 623 NFE setting in Restart sampler (Xu et al., 2023a) on ImageNet-64, for DisCo-Diff.

D.3. Evaluation

For the evaluation, we follow the standard protocol and compute the Fréchet distance between 50000 generated samples and the training images.

E. Molecular Docking

In this appendix, we will introduce the task of molecular docking and some of the existing approaches to tackle it for readers who are not familiar with this field. Experienced readers may skip to Section E.3 where we start describing the details of our docking approach.

E.1. Task Overview

Molecular docking consists of finding the 3D structure that a protein (also referred to as receptor) and a small molecule (or ligand) take when binding. This is an important task in drug design because most drugs are small molecules that operate by binding to a specific protein of interest in our body and inhibiting or enhancing its function. The common particular instantiation of the docking problem that we consider is also referred to as rigid blind docking i.e. where we are given as input the correct protein structure (*rigid*) but are not provided any information about where the ligand will bind on the protein nor the conformations it will take (*blind*).

Ground truth data for training and testing is obtained through experimental methods, like X-ray crystallography, that, for each protein-ligand complex, allow to observe a particular pose that protein and ligand took when binding together inside of the crystal. Although there may be other poses that this particular protein and ligand may take when binding in a natural environment, methods are evaluated based on their capacity to retrieve the crystal pose. This accuracy is typically computed as the percentage of test complexes where the predicted structure of the ligand (also referred to as ligand pose) is within a root mean square distance (RMSD) of 2Å from the ground truth when aligning the protein structures.

E.2. Related Work

Traditional approaches tackled the task via a search-based paradigm where, given a scoring function, they would search over possible ligand poses with an optimization algorithm to find a global minimum (Halgren et al., 2004; Trott & Olson, 2010). Recently, deep learning methods have been trying to speed up this search process by generating poses directly through a neural network. Initial approaches (Stärk et al., 2022; Lu et al., 2022) used regression-based frameworks to predict the pose, but, although significantly faster, they did not outperform traditional methods.

Corso et al. (2023) argued that the issue with these regression-based approaches is their treatment of the uncertainty in the multimodal model posterior pose distribution. They also proposed DiffDock, a diffusion-based generative model to generate docked poses that was able to outperform previous methods, which we use as a starting point for the integration of our DisCo-Diff approach to diffusion.

Most deep learning approaches to docking model the data as a geometric graph or point cloud in 3D. The nodes of this graph are the (heavy) atoms of the ligand and, typically, the C-alpha carbon atoms of the protein backbone (sometimes full-atom representations are also used for the protein but these are less common for computational complexity reasons). These nodes are connected by edges in case of chemical bonds or pairwise distances below a certain cutoff. Neural architectures learn features over the nodes of this graph through a number of message passing layers, the geometric structure is encoded via invariant (e.g. relying only on distance embeddings, see Schütt et al. (2017)) or equivariant operations (Geiger & Smidt, 2022).

E.3. Latent Variables

We design each latent variable to take values indicating one of the nodes in the protein-ligand joint graph. Therefore the codebook size for the latent variable of any given protein-ligand complex is equal to the total number of nodes in the graph

i.e. the sum of the number of atoms in the ligand and the number of residues in the protein. With this choice, intuitively each latent variable will indicate one particular atom or residue involved in some key component of the protein-ligand interaction. For example, using two latents the model can learn to indicate a geometric contact between a pair of nodes in the final representation. Further note, each "codebook", when considered as a set of one-hot vectors indicating notes, has a permutation equivariance property with the nodes of the graph (because they are associated with node properties): if the nodes of the input graph are permuted each latent variable coming out of the encoder or autoregressive model will also have its codebook representation permuted.

E.4. Architecture Details

Denoiser. This design choice for the discrete latents codebooks fits very well with the preexisting DiffDock's denoiser architecture composed of equivariant message passing layers (Geiger & Smidt, 2022). Each latent variable is encoded in a binary label for each node which is set to zero for all nodes except the one indicated by the latent. These binary labels are concatenated to the initial node features while the rest of the denoiser is kept unchanged. With probability 0.1 during training we drop the latents, in this case, the binary labels are set to zero for all latents, and a learnable null-embedding is fed to all initial node features.

Encoder. The encoder and autoregressive models adopt very similar architectures to the denoiser with a few key distinctions. The encoder takes as input the ground truth pose of the ligand, learns features for each node through message passing, and finally m separate feedforward MLPs (where m is the number of latents) with a one-dimensional output are applied to each node representation. The concatenated outputs of each of these MLPs form the logit vectors for each of the latent variables which are passed through the Gumbel-Softmax discretization step.

Autoregressive model. Unlike the image synthesis experiments setting where the images are often generated with relatively vague conditioning information, for docking, we are interested in generating ligand poses conditioned on a particular protein (structure) and ligand. This conditioning information significantly influences the posterior pose distribution and consequently the learned latent variables. Therefore, we need to condition the autoregressive model on the protein structure and the ligand. We achieve this, once again, through an equivariant message passing network, operating on an input composed of the protein structure and the ligand. The latter is centered at the protein's center, given an arbitrary conformer (i.e. molecular conformation) from RDKit (Landrum, 2013) and a uniformly random orientation. Like the denoiser, the autoregressive model takes as input the additional binary node labels for the existing latents (masked out appropriately during training), and, like the encoder, it uses its final node embeddings to predict the logits for the next latent variable.

E.5. Experimental Details

For the docking experiments, we follow the datasets and procedures established by Stärk et al. (2022) and Corso et al. (2023). Data for training and evaluation comes from the PDBBind dataset (Liu et al., 2017) with time-based splits (complexes before 2019 for training and validation, selected complexes from 2019 for testing).

Denoiser. We use a denoiser architecture analogous to the one proposed by Corso et al. (2024), which is a smaller version of DiffDock's original architecture where the main changes are: (1) 5 convolutional layers (vs 6 of the original DiffDock's architecture) (2) node representations with 24 scalars and pseudoscalars and 6 vectors and pseudovectors (vs respectively 48 and 10) (3) spherical harmonics order limited to 1 (vs 2). These changes, although somewhat affecting the inference quality, make training and testing of the models significantly more affordable (from 18 days on 4 GPUs to 9 days on 2 GPUs for training).

We keep the same denoiser architecture for both the baseline without discrete latents (DiffDock-S) and our model and apply similar hyperparameter searches when applicable to both models. At inference time, similarly to Corso et al. (2023), we take 40 independent samples and use the original DiffDock's confidence model[1] to select the top one. For DisCo-DiffDock each of the samples is taken by independently sampling from the autoregressive model and then the (conditioned) denoiser.

Encoder. For the encoder, we use a similar but slightly smaller architecture with 3 convolutional layers, 24 scalars, and 4 vectors. We set the number of discrete latent variables (each taking values over the whole set of possible nodes in the joint graph) to two, as we found this to equilibrate the complexity of the generative task between the score and autoregressive models.

[1]The confidence model is an additional model, Corso et al. (2023) trained to select the most likely correct poses out of the diffusion models samples. The reader can think of this as trying to select the maximum likelihood pose.

Autoregressive model. One challenge with the autoregressive model in this domain is its tendency to overfit the latent variables in the training set given the limited training data, the complexity of the conditioning information, and the low training signal that discrete labels provide. Therefore we found it beneficial to design the autoregressive model to use the pretrained layers of the denoiser itself. In particular, we simply add independent MLPs for each latent variable that are applied to the final scalar representations of the nodes. During the autoregressive training, for the first five epochs, the weights of the convolutional layers are frozen.

Inference hyperparameters. For inference, we maintain the number of inference steps from DiffDock (20) and, for both DisCo-DiffDock and the baseline, we tune on the validation set the sampling temperature for the different components of the diffusion similarly to how it was done by Ketata et al. (2023). For DisCo-DiffDock we also tune the temperature used to sample the autoregressive model. We find, with 40 samples, to be beneficial to set this temperature > 1 while the diffusion sampling temperature < 1, this corresponds to encouraging exploration of different binding modes while trying to obtain the maximum likelihood pose for each mode. This further highlights the advantage provided by enabling the decomposition of different degrees of uncertainty. Please see the source code in the Supplementary Material for all low level details and hyperparameters used.

F. Gaussian Mixture Experiments

F.1. Data Generation

For the toy example in section 3, we set the true data distribution to a mixture of eight Gaussian components:

$$p_{data}(\mathbf{x}) = \frac{1}{8}\sum_{i=1}^{8}\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_i, \sigma_i \boldsymbol{I}_{2\times 2})$$

where $\forall i, \sigma_i = 0.2$, and

$$\boldsymbol{\mu}_1 = \begin{pmatrix}3\\0\end{pmatrix}, \quad \boldsymbol{\mu}_2 = \begin{pmatrix}-3\\0\end{pmatrix}, \quad \boldsymbol{\mu}_3 = \begin{pmatrix}0\\3\end{pmatrix}, \quad \boldsymbol{\mu}_4 = \begin{pmatrix}0\\-3\end{pmatrix},$$
$$\boldsymbol{\mu}_5 = \begin{pmatrix}\frac{3}{\sqrt{2}}\\\frac{3}{\sqrt{2}}\end{pmatrix}, \quad \boldsymbol{\mu}_6 = \begin{pmatrix}\frac{3}{\sqrt{2}}\\\frac{-3}{\sqrt{2}}\end{pmatrix}, \quad \boldsymbol{\mu}_7 = \begin{pmatrix}\frac{3}{\sqrt{2}}\\\frac{-3}{\sqrt{2}}\end{pmatrix}, \quad \boldsymbol{\mu}_8 = \begin{pmatrix}\frac{-3}{\sqrt{2}}\\\frac{-3}{\sqrt{2}}\end{pmatrix}.$$

To construct the toy dataset, we randomly sampled 1000 data points from each component, totaling 8000 data points. We visualize the KDE plot of the generated data in Fig. 3(a).

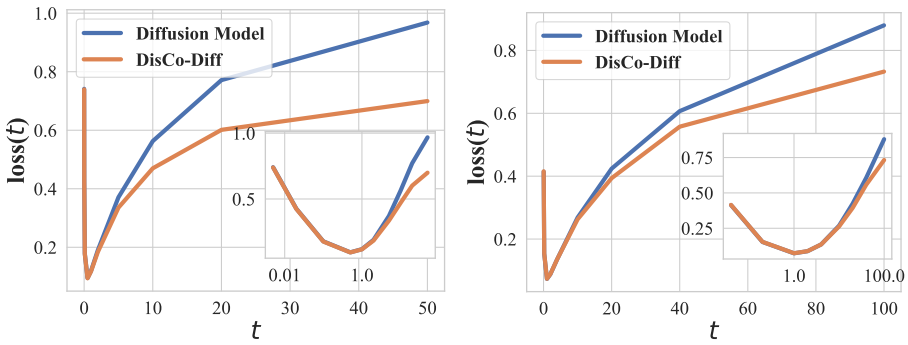
F.2. Training and Sampling

We employ a four-layer MLP as the diffusion decoder (Denoiser Neural Network) $\mathcal{G}$, for both Disco-Diff and diffusion models. We use a three-layer MLP as the encoder $\mathcal{E}$ in Disco-Diff. We set the latent dimension of discrete latent to 1 and the vocabulary size to 8. Ideally, each discrete latent should correspond to a Gaussian component, and the time-dependent scores for a single Gaussian component have a simple analytical expression. We leverage this simplicity and reparameterize the output of diffusion decoder as $\mathcal{G}(\mathbf{x}, t, z) = \frac{\mathcal{F}(z)-\mathbf{x}}{t^2+\sigma_1^2} + \mathcal{H}(\mathbf{x}, t)$, where $\mathcal{F}$ is the embedding for each discrete latent z and $\mathcal{H}$ is a four-layer MLP. The model optimization uses the Adam optimizer with a learning rate of 1e-3.

For sampling, we use the Heun's second-order sampler. We followed the time discretization scheme in EDM (Karras et al., 2022) with 50 sampling steps.

F.3. Metric

We detail the metrics used in Fig. 4. The curvature for points $\mathbf{x}(t)$ on ODE trajectory $d\mathbf{x}/dt = \mathcal{G}(\mathbf{x}, t, z)$ (z is null in diffusion models) is defined as $\kappa(\mathbf{x}(t)) = \frac{||\partial_t \boldsymbol{T}(\mathbf{x}(t),t)||}{||\mathbf{x}'(t)||}$ where $\boldsymbol{T}(\mathbf{x}, t) = \frac{\mathcal{G}(\mathbf{x}(t),t,z)}{||\mathcal{G}(\mathbf{x}(t),t,z)||}$ is the unit tangent vector. We can approximate the curvature by finite difference: $\kappa(\mathbf{x}(t)) = \frac{||\partial_t \boldsymbol{T}(t)||}{||\mathbf{x}'(t)||} \approx \frac{||\boldsymbol{T}(t)-\boldsymbol{T}(t-\Delta t)||}{||\mathbf{x}(t)-\mathbf{x}(t-\Delta t)||}$. We approximate $\mathbf{x}(t-\Delta t)$ by a single Euler step, *i.e.,* $\mathbf{x}(t-\Delta t) = \mathbf{x}(t) - \mathcal{G}(\mathbf{x}, t, z)\Delta t$. In Fig. 4(a), we report the expected curvature given the backward time when simulating the ODE, *i.e.,* $\mathbb{E}_{\mathbf{x}(t)}\left[\frac{||\boldsymbol{T}(t)-\boldsymbol{T}(t-\Delta t)||}{||\mathbf{x}(t)-\mathbf{x}(t-\Delta t)||}\right]$. We set the time elapsed to $\Delta t = 0.001$.



(a) ImageNet-64 (b) ImageNet-128

Figure 10. Averaged training loss versus noise level t.

In Fig. 4(b), we measure the complexity of the trained neural networks using the expected squared Frobenius norm of the network's Jacobians, *i.e.,* $\mathbb{E}_{\mathbf{x}(t)}\left[||\nabla_{\mathbf{x}}\mathcal{G}(\mathbf{x},t,z)||_F^2\right]$.

Additionally, to quantitatively evaluate the generation quality, we report the Wasserstein-2 (W-2) distance between the generated distribution and the ground truth distribution. In the DisCo-Diff model, the W-2 distance is at 0.118, compared to 0.27 in the standard diffusion model. It suggests that DisCo-Diff better captures the multimodal distribution, even in 2-dim space.

G. Additional Samples and Experiments

G.1. Loss Analysis

In Fig. 10, we provide the loss versus time curve on both ImageNet-64 and ImageNet-128 datasets. We have also included a log-scale version of the x-axis in the inset plot.

G.2. Class-conditoned ImageNet-64

We provide extended samples generated by DisCo-Diff in Fig. 11.

G.3. Class-conditoned ImageNet-128

We provide extended samples generated by DisCo-Diff in Fig. 12. We also visualize samples with shared discrete latents in Fig. 14. We further provide samples using different samplers in Fig. 13, to highlight the over-smoothing issue of DisCo-Diff at smaller times we observed for this model when using the DDPM sampler. Although, in theory, the VDM++ model and DisCo-Diff should learn the same field at small times, in practice, we observe that VDM++ DDPM provides samplers with richer details compared to DisCo-Diff DDPM at smaller times. It supports our hypothesis that the discrete latents tend to divert the model to overlook the high-level details on ImageNet-128, when using the DDPM sampler and the network architecture in (Kingma & Gao, 2023). We would like to emphasize that we only observed this behavior for this one model and dataset. In all other experiments, discrete latents universally improved performance for all stochastic and non-stochastic samplers, even when used for all times t.

G.4. Group Hierarchical DisCo-Diff

We further provide extended samples from the Group hierarchical DisCo-Diff. Fig. 15 showcases the generated images when composing two discrete latents together, *i.e.,* $(\mathbf{z}_{0:20}, \hat{\mathbf{z}}_{20:30})$. We can see that the generated images from composed latent generally inherit the shape from images generated by $\mathbf{z}$, and the color from images generated by $\hat{\mathbf{z}}$.

Fig. 16 further shows the effect when progressively fixing more coordinates of the discrete latent, and sampling the remaining coordinates by the auto-regressive model. The images first converge in shape/layout, and subsequently converge in color/texture.

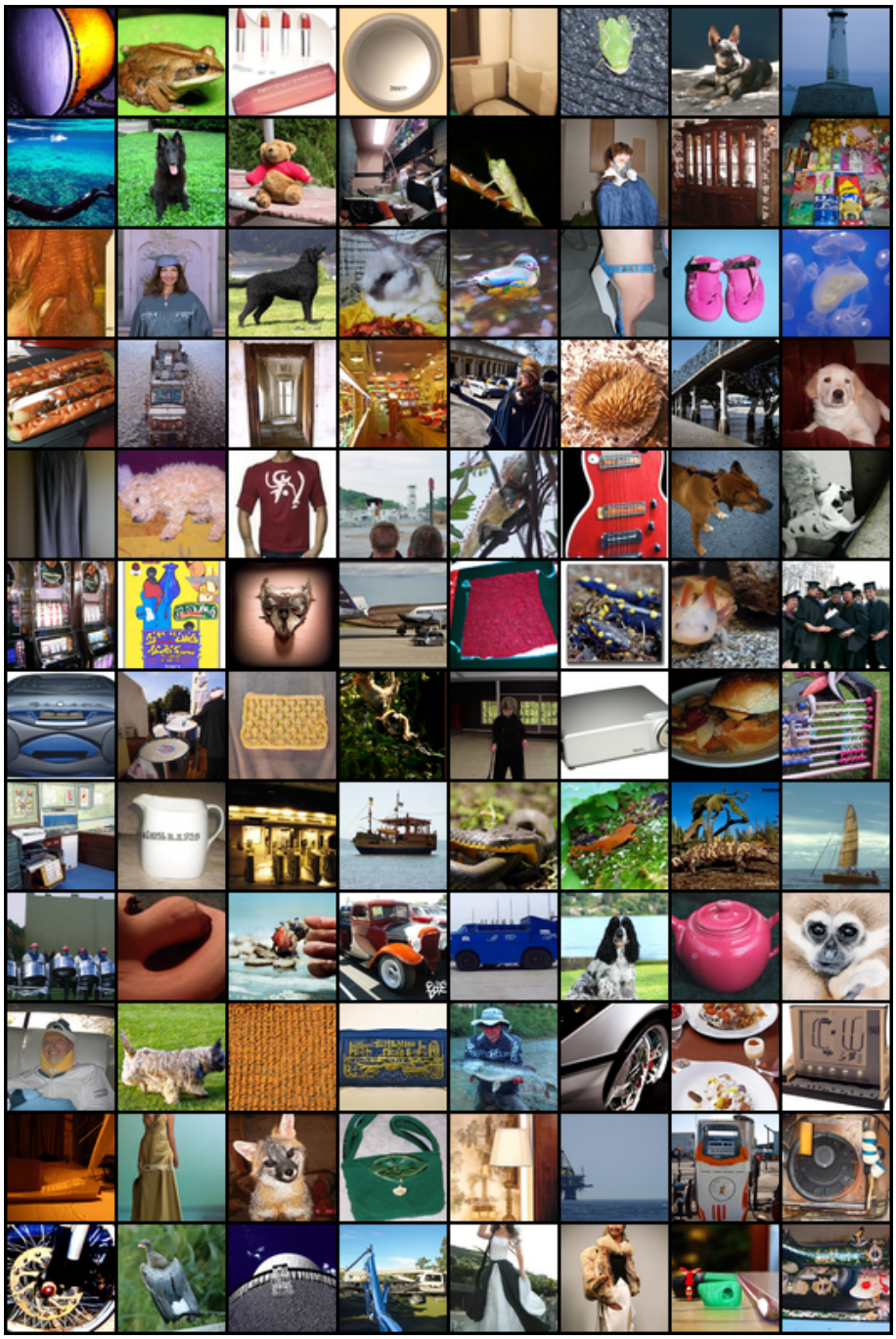


Figure 11. Generated samples by DisCo-Diff on class-conditioned ImageNet-64, with ODE sampler (FID=1.65, NFE=78).

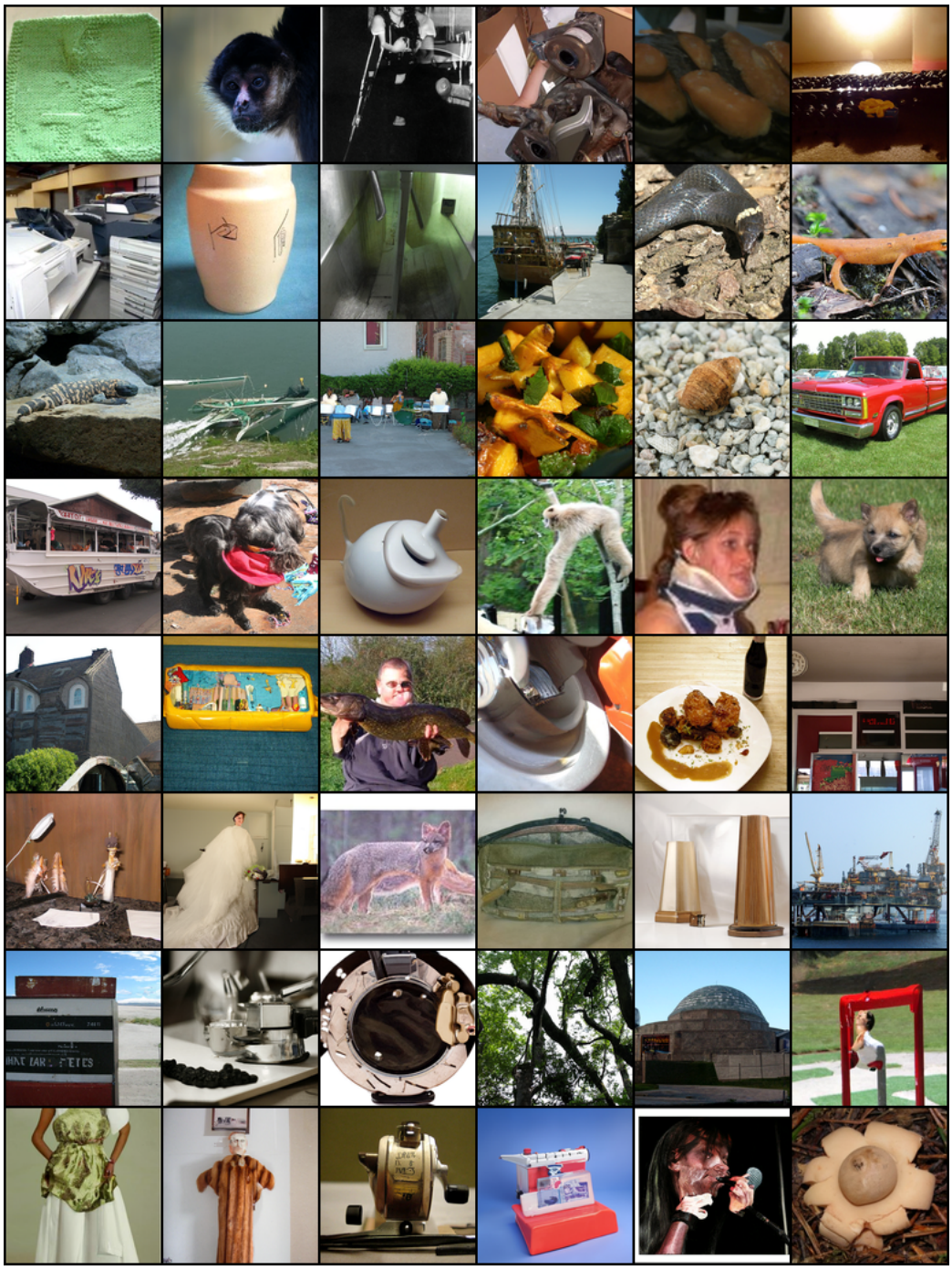
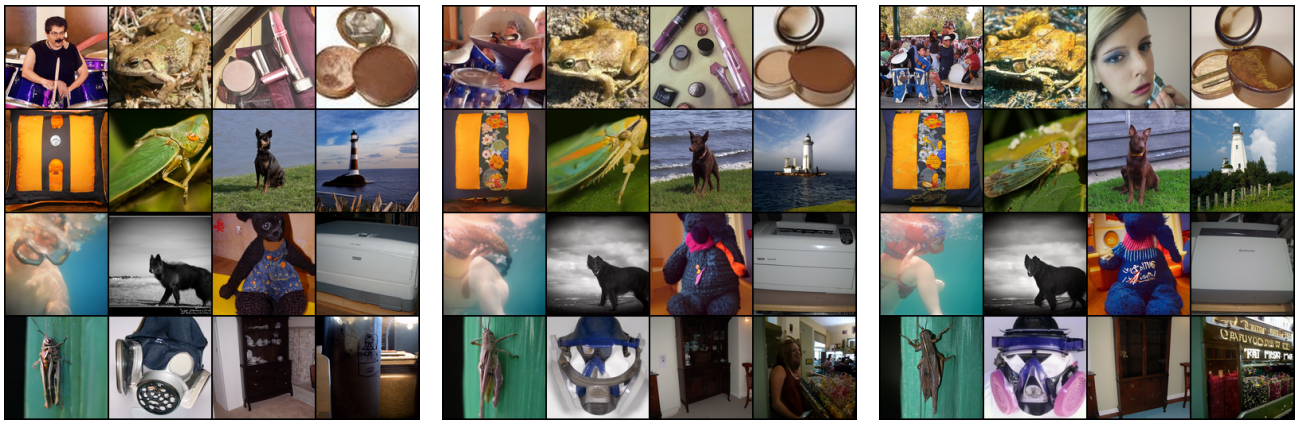


Figure 12. Generated samples by DisCo-Diff on class-conditioned ImageNet-128, with ODE sampler (FID=2.08, NFE=114).



(a) DisCo-Diff ODE, FID=1.98

(b) DisCo-Diff ODE ($t \geq 10$) and DisCo-Diff DDPM ($t < 10$), , FID=2.78

(c) DisCo-Diff ODE ($t \geq 10$) and VDM DDPM ($t < 10$), FID=1.73

Figure 13. Images generated by different samplers. When using the DDPM sampler at smaller times (b), the generated images exhibit a slight over-smoothing issue, losing some high-frequency details in comparison to those produced with the VDM++ DDPM sampler at smaller times. Note that FID score typically penalizes over-smooth samples. This observation supports our hypothesis that the use of the DDPM sampler in DisCo-Diff at smaller times can in certain situations overlook high-frequency details.

G.5. Overfitting and Encoder Collapse in Continuous Latent

In this section, we show that it is difficult to control the amount of information stored in the continuous latents, which would lead to either overfitting or encoder collapse. To illustrate the issue, we derived the continuous / discrete latents from a specific real image, and fed the corresponding latents into the denoisers. As shown in Figure 17, when KLD weight=0.1, the continuous latent model exhibits the overfitting issue, as all the generated images are very similar to the training image. It also indicates that the encoder squeezes excessive information in the continuous latent when using a smaller KLD weight, which complicates generative training in the second stage. When KLD weight=1, the model exhibits encoder collapse – the denoiser would ignore the continuous latent. We observe that even using a different continuous latent, the model will still generate an identical batch of samples. In contrast, DisCo-Diff generated a batch of diverse samples, sharing a similar high-level layout and color with the training image. This indicates that the discrete latents in DisCo-Diff encode global layout and color attributes — key statistical elements crucial for the diffusion process in Euclidean space. This aligns with more direct and straighter ODE trajectories.

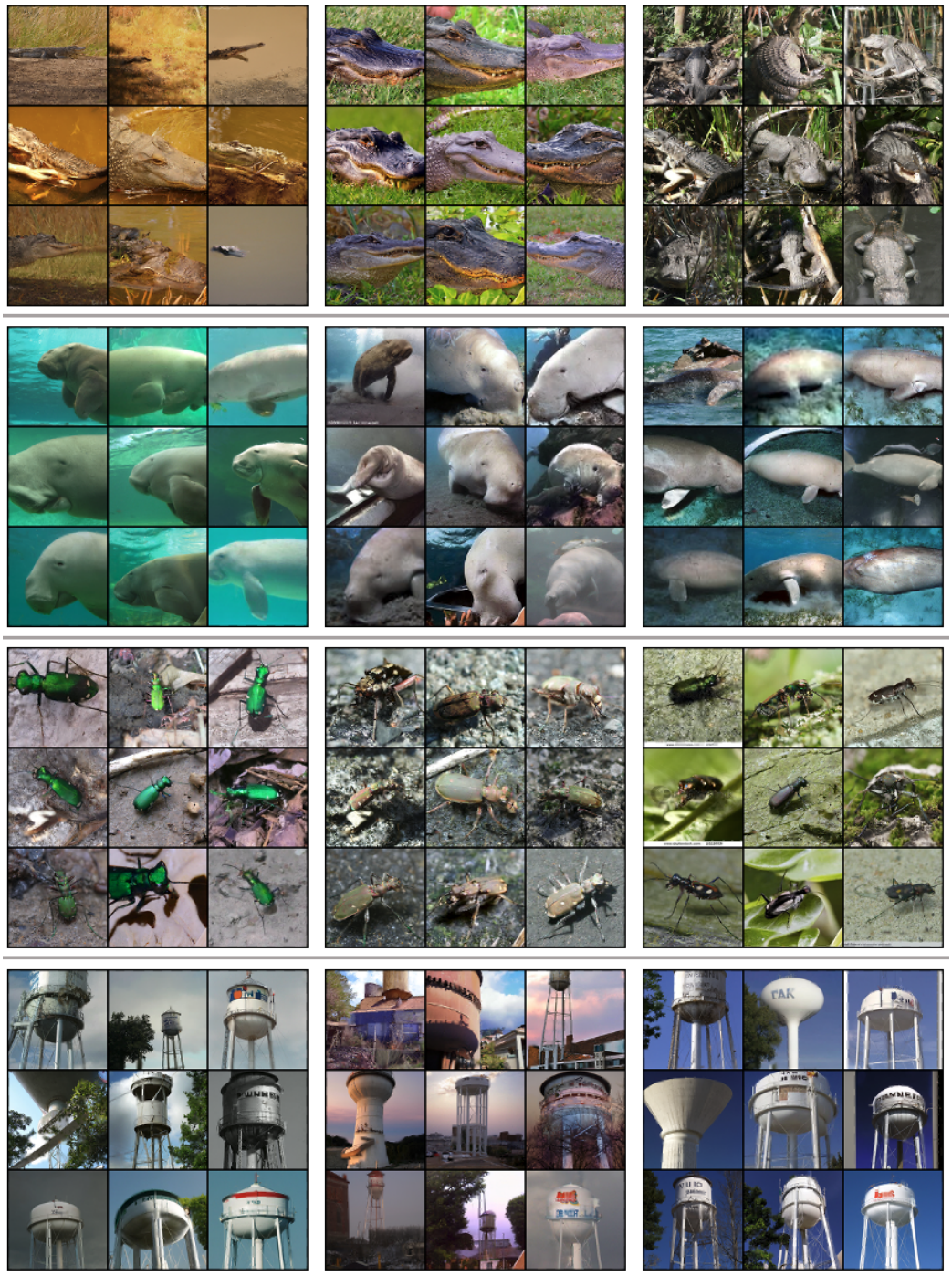


Figure 14. Generated samples by DisCo-Diff on class-conditioned ImageNet-128, with ODE sampler. Samples in each grid share the same latent, and grids in each row share the same class labels. We can see that generally, images sharing the same discrete latents demonstrate similar global characteristics, such as shape, layout, and color, despite being under the same class. It suggests that discrete latents provide complementary information to the class labels.

Figure 15. Generated images with a shared latent, using group hierarchical DisCo-Diff trained on ImageNet-64. *Left:* Shared latent $\mathbf{z}$. *Middle:* Shared latent $\hat{\mathbf{z}}$. *Right:* Shared latent $(\mathbf{z}_{0:20}, \hat{\mathbf{z}}_{20:30})$, where the first 20 coordinates are from $\mathbf{z}$ and the last 10 coordinates are from $\hat{\mathbf{z}}$. We can see that the generated images from composed latents generally inherit the shape from images generated by $\mathbf{z}$, and the color from images generated by $\hat{\mathbf{z}}$.

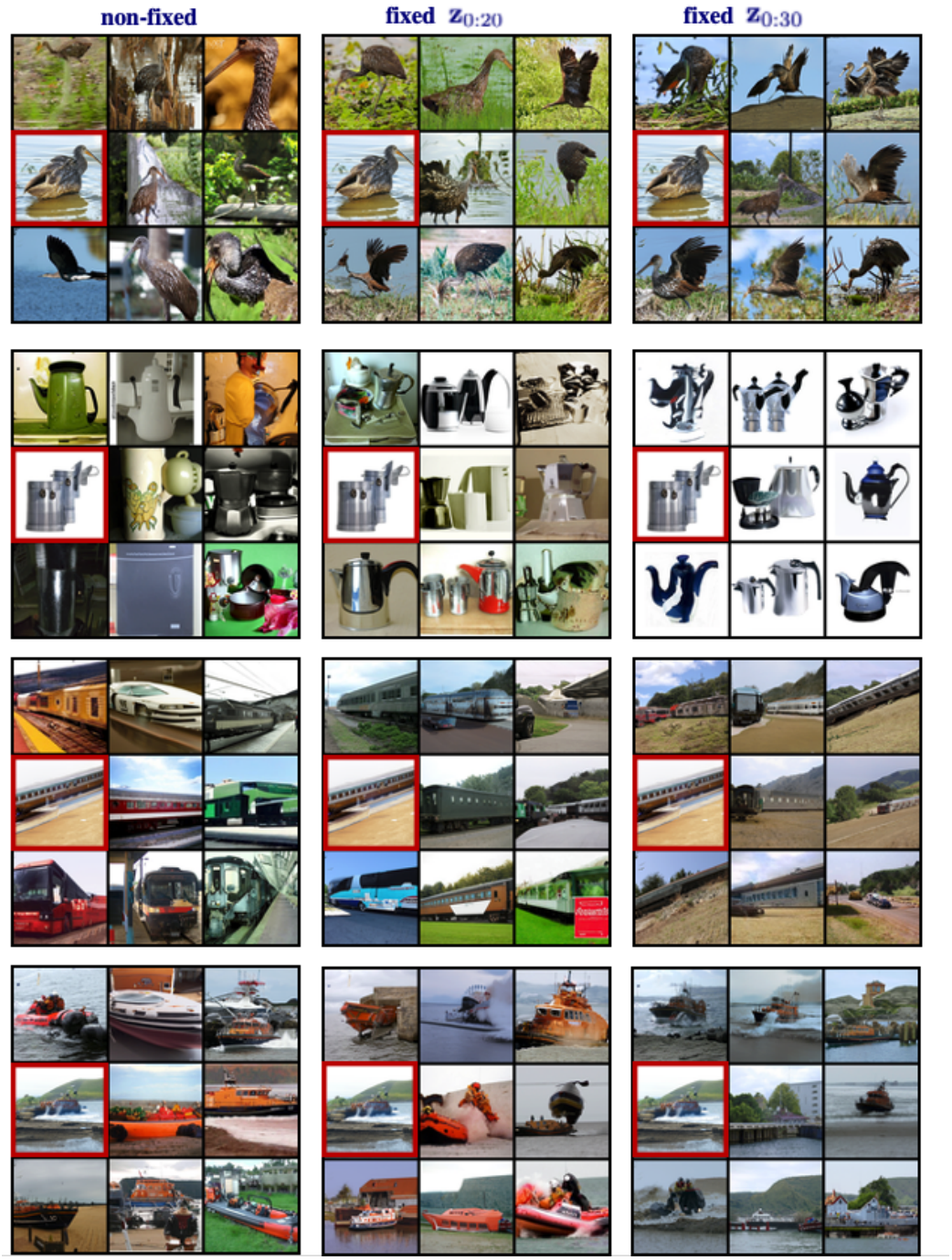


Figure 16. Progressively fixing more subcoordinates of the discrete latents, using our group hierarchical DisCo-Diff on ImageNet-64. *Left:* Randomly sampled $\mathbf{z}$. *Middle:* Fixing the first 20 coordinates $\mathbf{z}_{:20}$ as the one derived from the red-boxed image, sampling the rest. *Right:* Fixing the whole 30-dim. $\mathbf{z}$ as the one derived from the red-boxed image. The figure shows the effect when progressively fixing more coordinates of the discrete latent, and sampling the remaining coordinates by the auto-regressive model. The images first converge in shape/layout, and subsequently converge in color/texture.

(a) Real image (b) KLD weight=0.1 (c) KLD weight=1 (d) DisCo-Diff

Figure 17. We derived the continuous / discrete latents from the real image in (a) and fed the latents into the denoisers. (b): The continuous latent model exhibits the overfitting issue when KLD weight equals to 0.1, as all the generated images are very similar to the real image. (c): When applying a stronger KL regulation (KLD weight=1) on the continuous latent, the model exhibits encoder collapse – the denoiser would ignore the continuous latent. (d): DisCo-Diff generated a batch of diverse samples, sharing a similar high-level layout and color with the real image.