Diffusion bridges vector quantized Variational AutoEncoders

Max Cohen, Guillaume Quispe, Sylvain Le Corff, Charles Ollion, Eric Moulines

Diffusion bridges vector quantized Variational AutoEncoders

Max Cohen[⋆,‡], Guillaume Quispe[†], Sylvain Le Corff[⋆], Charles Ollion[†], and Éric Moulines[†]

[⋆]SAMOVAR, Télécom SudParis, Institut Polytechnique de Paris, Palaiseau.
[‡]Accenta, Boulogne-Billancourt.
[†]CMAP, École Polytechnique, Institut Polytechnique de Paris, Palaiseau.

Abstract

Vector Quantized-Variational AutoEncoders (VQ-VAE) are generative models based on discrete latent representations of the data, where inputs are mapped to a finite set of learned embeddings. To generate new samples, an autoregressive prior distribution over the discrete states must be trained separately. This prior is generally very complex and leads to slow generation. In this work, we propose a new model to train the prior and the encoder/decoder networks simultaneously. We build a diffusion bridge between a continuous coded vector and a non-informative prior distribution. The latent discrete states are then given as random functions of these continuous vectors. We show that our model is competitive with the autoregressive prior on the mini-Imagenet and CIFAR dataset and is efficient in both optimization and sampling. Our framework also extends the standard VQ-VAE and enables end-to-end training.

1 Introduction

Variational AutoEncoders (VAE) have emerged as important generative models based on latent representations of the data. While the latent states are usually continuous vectors, Vector Quantized Variational AutoEncoders (VQ-VAE) have demonstrated the usefulness of discrete latent spaces and have been successfully applied in image and speech generation [Oord et al., 2017, Esser et al., 2021, Ramesh et al., 2021].

In a VQ-VAE, the distribution of the inputs is assumed to depend on a hidden discrete state. Large scale image generation VQ-VAEs use for instance multiple discrete latent states, typically organized as 2-dimensional lattices. In the original VQ-VAE, the authors propose a variational approach to approximate the posterior distribution of the discrete states given the observations. The variational distribution takes as input the observation, which is passed through an encoder. The discrete latent variable is then computed by a nearest neighbour procedure that maps the encoded vector to the nearest discrete embedding.

It has been argued that the success of VQ-VAEs lies in the fact that they do not suffer from the usual posterior collapse of VAEs [Oord et al., 2017]. However, the implementation of VQ-VAE involves many practical tricks and still suffers from several limitations. First, the quantization step leads the authors to propose a rough approximation of the gradient of the loss function by copying gradients from the decoder input to the encoder output. Second, the prior distribution of the discrete variables is initially assumed to be uniform when training the VQ-VAE. In a second training step, high-dimensional autoregressive models such as PixelCNN [van den Oord et al., 2016, Salimans et al., 2017, Chen et al., 2018] and WaveNet [Oord et al., 2016] are estimated to obtain a complex prior distribution. Joint training of the prior and the VQ-VAE is a challenging task for which no satisfactory solutions exist yet. Our work addresses both problems by introducing a new

mathematical framework that extends and generalizes the standard VQ-VAE. Our method enables end-to-end training and, in particular, bypasses the separate training of an autoregressive prior.

An autoregressive pixelCNN prior model has several drawbacks, which are the same in the pixel space or in the latent space. The data is assumed to have a fixed sequential order, which forces the generation to start at a certain point, typically in the upper left corner, and span the image or the 2-dimensional latent lattice in an arbitrary way. At each step, a new latent variable is sampled using the previously sampled pixels or latent variables. Inference may then accumulate prediction errors, while training provides ground truth at each step. The runtime process, which depends mainly on the number of network evaluations, is sequential and depends on the size of the image or the 2-dimensional latent lattice, which can become very large for high-dimensional objects.

The influence of the prior is further explored in [Razavi et al., 2019], where VQ-VAE is used to sample images on a larger scale, using two layers of discrete latent variables, and [Willetts et al., 2021] use hierarchical discrete VAEs with numerous layers of latent variables. Other works such as [Esser et al., 2021, Ramesh et al., 2021] have used Transformers to autoregressively model a sequence of latent variables: while these works benefit from the recent advances of Transformers for large language models, their autoregressive process still suffers from the same drawbacks as pixelCNN-like priors.

The main claim of our paper is that using diffusions in a continuous space, $\mathbb{R}^{d\times N}$ in our setting, is a very efficient way to learn complex discrete distributions, with support on a large space (here with cardinality K^N). We only require an embedded space, an uninformative target distribution (here a Gaussian law), and use a continuous bridge process to learn the discrete target distribution. In that direction, our contribution is inspired by the literature but also significantly different. Our procedure departs from the diffusion probabilistic model approach of [Ho et al., 2020], which highlights the role of bridge processes in denoising continuous target laws, and from [Hoogeboom et al., 2021], where multinomial diffusions are used to noise and denoise but prevent the use of the expressiveness of continuous bridges, and also do not scale well with K as remarked by its authors. Although we target a discrete distribution, our approach does not suffer from this limitation.

Our contributions are summarized as follows.

- We propose a new mathematical framework for VQ-VAEs. We introduce a two-stage prior distribution. Following the diffusion probabilistic model approach of [Ho et al., 2020], we consider first a continuous latent vector parameterized as a Markov chain. The discrete latent states are defined as random functions of this Markov chain. The transition kernels of the continuous latent variables are trained using diffusion bridges to gradually produce samples that match the data.
- To our best knowledge, this is the first probabilistic generative model to use denoising diffusion in discrete latent space. This framework allows for end-to-end training of VQ-VAE.
- We focus on VQ-VAE as our framework enables simultaneous training of all components of those popular discrete models which is not straightforward. However, our methodology is more general and allows the use of continuous embeddings and diffusion bridges to sample form any discrete laws.
- We present our method on a toy dataset and then compare its efficiency to the pixelCNN prior of the original VQ-VAE on the miniImagenet dataset.

Figure 1 describes the complete architecture of our model.

2 Related Works

Diffusion Probabilistic Models. A promising class of models that depart from autoregressive models are Diffusion Probabilistic Models [Sohl-Dickstein et al., 2015, Ho et al., 2020] and closely related Score-Matching Generative Models [Song and Ermon, 2019, De Bortoli et al., 2021]. The general idea is to apply

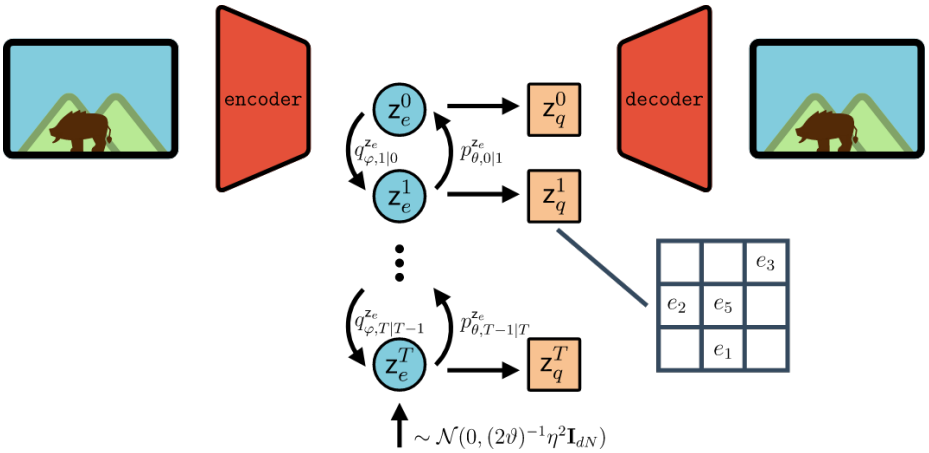

Figure 1: Our proposed architecture, for a prior based on a Ornstein-Uhlenbeck bridge. The top pathway from *input image* to z_e^0, to z_q^0, to *reconstructed image* resembles the original VQ-VAE model. The vertical pathway from $(\mathsf{z}_e^0, \mathsf{z}_q^0)$ to $(\mathsf{z}_e^T, \mathsf{z}_q^T)$ and backwards is based on a denoising diffusion process. See Section 3.2 and Algorithm 2 for the corresponding sampling procedure.

a corrupting Markovian process on the data through T corrupting steps and learn a neural network that gradually *denoises* or reconstructs the original samples from the noisy data. For example, when sampling images, an initial sample is drawn from an uninformative distribution and reconstructed iteratively using the trained Markov kernel. This process is applied to all pixels simultaneously, so no fixed order is required and the sampling time does not depend on sequential predictions that depend on the number of pixels, but on the number of steps T. While this number of steps can be large ($T = 1000$ is typical), simple improvements enable to reduce it dramatically and obtain $\times 50$ speedups [Song et al., 2021]. These properties have led diffusion probability models to receive much attention in the context of continuous input modelling.

From Continuous to Discrete Generative denoising. In [Hoogeboom et al., 2021], the authors propose multinomial diffusion to gradually add categorical noise to discrete samples for which the generative denoising process is learned. Unlike alternatives such as normalizing flows, the diffusion proposed by the authors for discrete variables does not require gradient approximations because the parameter of the diffusion is fixed.

Such diffusion models are optimized using variational inference to learn the denoising process, i.e., the bridge that aims at inverting the multinomial diffusion. In [Hoogeboom et al., 2021], the authors propose a variational distribution based on bridge sampling. In [Austin et al., 2021], the authors improve the idea by modifying the transition matrices of the corruption scheme with several tricks. The main one is the addition of absorbing states in the corruption scheme by replacing a discrete value with a MASK class, inspired by recent Masked Language Models like BERT. In this way, the corrupted dimensions can be distinguished from the original ones instead of being uniformly sampled. One drawback of their approach, mentioned by the authors, is that the transition matrix does not scale well for a large number of embedding vectors, which is typically the case in VQ-VAE.

Compared to discrete generative denoising, our approach takes advantage of the fact that the discrete distribution depends solely on a continuous distribution in VQ-VAE. We derive a novel model based on continuous-discrete diffusion that we believe is simpler and more scalable than the models mentioned in this section.

From Data to Latent Generative denoising. Instead of modelling the data directly, [Vahdat et al., 2021] propose to perform score matching in a latent space. The authors propose a complete generative model and are able to train the encoder/decoder and score matching end-to-end. Their method also achieve excellent visual patterns and results but relies on a number of optimization heuristics necessary for stable training. In [Mittal et al., 2021], the authors have also applied such an idea in a generative music model. Instead of working in a continuous latent space, our method is specifically designed for a discrete latent space as in VQ-VAEs.

Using Generative denoising in discrete latent space. In the model proposed by [Gu et al., 2021], the autoregressive prior is replaced by a discrete generative denoising process, which is perhaps closer to our idea. However, the authors focus more on a text-image synthesis task where the generative denoising model is traine based on an input text: it generates a set of discrete visual tokens given a sequence of text tokens. They also consider the VQ-VAE as a trained model and focus only on the generation of latent variables. This work focuses instead on deriving a full generative model with a sound probabilistic interpretation that allows it to be trained end-to-end.

3 Diffusion bridges VQ-VAE

3.1 Model and loss function

Assume that the distribution of the input $x \in \mathbb{R}^m$ depends on a hidden discrete state $\mathsf{z}_q \in \mathcal{E} = \{\mathrm{e}_1, \dots, \mathrm{e}_K\}$ with $\mathrm{e}_k \in \mathbb{R}^d$ for all $1 \leqslant k \leqslant K$. Let p_θ be the joint probability density of (z_q, x)

$$(\mathsf{z}_q, x) \mapsto p_\theta(\mathsf{z}_q, x) = p_\theta(\mathsf{z}_q) p_\theta(x|\mathsf{z}_q)\,,$$

where $\theta \in \mathbb{R}^p$ are unknown parameters. Consider first an encoding function f_φ and write $\mathsf{z}_e(x) = f_\varphi(x)$ the encoded data. In the original VQ-VAE, the authors proposed the following variational distribution to approximate $p_\theta(\mathsf{z}_q|x)$:

$$q_\varphi(\mathsf{z}_q|x) = \delta_{\mathrm{e}_{k_x^*}}(\mathsf{z}_q)\,,$$

where δ is the Dirac mass and

$$k_x^* = \mathrm{argmin}_{1 \leqslant k \leqslant K} \left\{ \|\mathsf{z}_e(x) - \mathrm{e}_k\|_2 \right\}\,,$$

where $\varphi \in \mathbb{R}^r$ are all the variational parameters.

In this paper, we introduce a diffusion-based generative VQ-VAE. This model allows to propose a VAE approach with an efficient joint training of the prior and the variational approximation. Assume that z_q is a sequence, i.e. $\mathsf{z}_q = \mathsf{z}_q^{0:T}$, where for all sequences $(a_u)_{u \geqslant 0}$ and all $0 \leqslant s \leqslant t$, $a^{s:t}$ stands for $(a_s, \dots, a_t)$. Consider the following joint probability distribution

$$p_\theta(\mathsf{z}_q^{0:T}, x) = p_\theta^{\mathsf{z}_q}(\mathsf{z}_q^{0:T}) p_\theta^x(x|\mathsf{z}_q^0)\,.$$

The latent discrete state z_q^0 used as input in the decoder is the final state of the chain $(\mathsf{z}_q^T, \dots, \mathsf{z}_q^0)$. We further assume that $p_\theta^{\mathsf{z}_q}(\mathsf{z}_q^{0:T})$ is the marginal distribution of

$$p_\theta(\mathsf{z}_q^{0:T}, \mathsf{z}_e^{0:T}) = p_{\theta,T}^{\mathsf{z}_e}(\mathsf{z}_e^T) p_{\theta,T}^{\mathsf{z}_q}(\mathsf{z}_q^T|\mathsf{z}_e^T) \prod_{t=0}^{T-1} p_{\theta,t|t+1}^{\mathsf{z}_e}(\mathsf{z}_e^t|\mathsf{z}_e^{t+1}) p_{\theta,t}^{\mathsf{z}_q}(\mathsf{z}_q^t|\mathsf{z}_e^t)\,.$$

In this setting, $\{\mathsf{z}_e^t\}_{0 \leqslant t \leqslant T}$ are continuous latent states in $\mathbb{R}^{d \times N}$ and conditionally on $\{\mathsf{z}_e^t\}_{0 \leqslant t \leqslant T}$ the $\{\mathsf{z}_q^t\}_{0 \leqslant t \leqslant T}$ are independent with discrete distribution with support $\mathcal{E}^N$. This means that we model jointly N latent states

as this is useful for many applications such as image generation. The continuous latent state is assumed to be a Markov chain and at each time step t the discrete variable z_q^t is a random function of the corresponding z_e^t. Although the continuous states are modeled as a Markov chain, the discrete variables arising therefrom have a more complex statistical structure (and in particular are not Markovian).

The prior distribution of z_e^T is assumed to be uninformative and this is the sequence of denoising transition densities $\{p^{\mathsf{z}_e}_{\theta,t|t+1}\}_{0\leqslant t\leqslant T-1}$ which provides the final latent state z_e^0 which is mapped to the embedding space and used in the decoder, i.e. the conditional law of the data given the latent states. The final discrete z_q^0 only depends the continuous latent variable z_e^0, similar to the dependency between z_q and z_e in the original VQ-VAE.

Since the conditional law $p_\theta(\mathsf{z}_q^{0:T}, \mathsf{z}_e^{0:T}|x)$ is not available explicitly, this work focuses on variational approaches to provide an approximation. Then, consider the following variational family:

$$q_\varphi(\mathsf{z}_q^{0:T}, \mathsf{z}_e^{0:T}|x) = \delta_{\mathsf{z}_e(x)}(\mathsf{z}_e^0) q^{\mathsf{z}_q}_{\varphi,0}(\mathsf{z}_q^0|\mathsf{z}_e^0) \prod_{t=1}^{T} \left\{ q^{\mathsf{z}_e}_{\varphi,t|t-1}(\mathsf{z}_e^t|\mathsf{z}_e^{t-1}) q^{\mathsf{z}_q}_{\varphi,t}(\mathsf{z}_q^t|\mathsf{z}_e^t) \right\} .$$

The family $\{q^{\mathsf{z}_e}_{\varphi,t|t-1}\}_{1\leqslant t\leqslant T}$ of forward "noising" transition densities are chosen to be the transition densities of a continuous-time process $(Z_t)_{t\geqslant 0}$ with $Z_0 = \mathsf{z}_e(x)$. Sampling the diffusion bridge $(\tilde{Z}_t)_{t\geqslant 0}$, i.e. the law of the process $(Z_t)_{t\geqslant 0}$ conditioned on $Z_0 = \mathsf{z}_e(x)$ and $Z_T = \mathsf{z}_e^T$ is a challenging problem for general diffusions, see for instance [Beskos et al., 2008, Lin et al., 2010, Bladt et al., 2016]. By the Markov property, the marginal density at time t of this conditioned process is given by:

$$\tilde{q}^{\mathsf{z}_e}_{\varphi,t|0,T}(\mathsf{z}_e^t|\mathsf{z}_e^0, \mathsf{z}_e^T) = \frac{q^{\mathsf{z}_e}_{\varphi,t|0}(\mathsf{z}_e^t|\mathsf{z}_e^0) q^{\mathsf{z}_e}_{\varphi,T|t}(\mathsf{z}_e^T|\mathsf{z}_e^t)}{q^{\mathsf{z}_e}_{\varphi,T|0}(\mathsf{z}_e^T|\mathsf{z}_e^0)} . \tag{1}$$

The Evidence Lower BOund (ELBO) is then defined, for all (θ, φ), as

$$\mathcal{L}(\theta, \varphi) = \mathbb{E}_{q_\varphi} \left[\log \frac{p_\theta(\mathsf{z}_q^{0:T}, \mathsf{z}_e^{0:T}, x)}{q_\varphi(\mathsf{z}_q^{0:T}, \mathsf{z}_e^{0:T}|x)} \right] ,$$

where $\mathbb{E}_{q_\varphi}$ is the expectation under $q_\varphi(\mathsf{z}_q^{0:T}, \mathsf{z}_e^{0:T}|x)$.

Lemma 3.1. *For all (θ, φ), the ELBO $\mathcal{L}(\theta, \varphi)$ is:*

$$\mathcal{L}(\theta, \varphi) = \mathbb{E}_{q_\varphi} \left[\log p^x_\theta(x|\mathsf{z}_q^0) \right] + \sum_{t=0}^{T} \mathcal{L}_t(\theta, \varphi) + \sum_{t=0}^{T} \mathbb{E}_{q_\varphi} \left[\log \frac{p^{\mathsf{z}_q}_{\theta,t}(\mathsf{z}_q^t|\mathsf{z}_e^t)}{q^{\mathsf{z}_q}_{\varphi,t}(\mathsf{z}_q^t|\mathsf{z}_e^t)} \right] ,$$

where, for $1 \leqslant t \leqslant T-1$,

$$\begin{aligned}
\mathcal{L}_0(\theta, \varphi) &= \mathbb{E}_{q_\varphi} \left[\log p^{\mathsf{z}_e}_{\theta,0|1}(\mathsf{z}_e^0|\mathsf{z}_e^1) \right] , \\
\mathcal{L}_t(\theta, \varphi) &= \mathbb{E}_{q_\varphi} \left[\log \frac{p^{\mathsf{z}_e}_{\theta,t-1|t}(\mathsf{z}_e^{t-1}|\mathsf{z}_e^t)}{q^{\mathsf{z}_e}_{\varphi,t-1|0,t}(\mathsf{z}_e^{t-1}|\mathsf{z}_e^0, \mathsf{z}_e^t)} \right] , \\
\mathcal{L}_T(\theta, \varphi) &= \mathbb{E}_{q_\varphi} \left[\log \frac{p^{\mathsf{z}_e}_{\theta,T}(\mathsf{z}_e^T)}{q^{\mathsf{z}_e}_{\varphi,T|0}(\mathsf{z}_e^T|\mathsf{z}_e^0)} \right] .
\end{aligned}$$

Proof. The proof is standard and postponed to Appendix 6. □

The three terms of the objective function can be interpreted as follows:

$$\mathcal{L}(\theta,\varphi) = \mathcal{L}^{rec}(\theta,\varphi) + \sum_{t=0}^{T} \mathcal{L}_t(\theta,\varphi) + \sum_{t=0}^{T} \mathcal{L}_t^{reg}(\theta,\varphi)$$

with $\mathcal{L}^{rec} = \mathbb{E}_{q_\varphi}[\log p_\theta^x(x|\mathsf{z}_q^0)]$ a reconstruction term, $\mathcal{L}_t$ the diffusion term, and an extra term

$$\mathcal{L}_t^{reg} = \mathbb{E}_{q_\varphi}\left[\log \frac{p_{\theta,t}^{\mathsf{z}_q}(\mathsf{z}_q^t|\mathsf{z}_e^t)}{q_{\varphi,t}^{\mathsf{z}_q}(\mathsf{z}_q^t|\mathsf{z}_e^t)}\right], \tag{2}$$

which may be seen as a regularization term as discussed in next sections.

3.2 Application to Ornstein-Uhlenbeck processes

Consider for instance the following Stochastic Differential Equation (SDE) to add noise to the normalized inputs:

$$\mathrm{d}Z_t = -\vartheta(Z_t - z_*)\mathrm{d}t + \eta \mathrm{d}W_t\,, \tag{3}$$

where $\vartheta, \eta > 0$, $z_* \in \mathbb{R}^{d\times N}$ is the target state at the end of the noising process and $\{W_t\}_{0\leqslant t\leqslant T}$ is a standard Brownian motion in $\mathbb{R}^{d\times N}$. We can define the variational density by integrating this SDE along small step-sizes. Let δ_t be the time step between the two consecutive latent variables z_e^{t-1} and z_e^t. In this setting, $q_{\varphi,t|t-1}^{\mathsf{z}_e}(\mathsf{z}_e^t|\mathsf{z}_e^{t-1})$ is a Gaussian probability density function with mean $z_* + (\mathsf{z}_e^{t-1} - z_*)\mathrm{e}^{-\vartheta\delta_t}$ in $\mathbb{R}^{d\times N}$ and covariance matrix $(2\vartheta)^{-1}\eta^2(1-\mathrm{e}^{-2\vartheta\delta_t})\mathbf{I}_{dN}$, where for all $n \geqslant 1$, $\mathbf{I}_n$ is the identity matrix with size $n\times n$. Asymptotically the process is a Gaussian with mean z_* and variance $\eta^2(2\vartheta)^{-1}\mathbf{I}_{dN}$.

The denoising process amounts then to sampling from the bridge associated with the SDE, i.e. sampling z_e^{t-1} given z_e^0 and z_e^t. The law of this bridge is explicit for the Ornstein-Uhlenbeck diffusion (3). Using (1),

$$\tilde{q}_{\varphi,s|0,t}^{\mathsf{z}_e}(\mathsf{z}_e^s|\mathsf{z}_e^t,\mathsf{z}_e^0) \propto q_{\varphi,s|0}^{\mathsf{z}_e}(\mathsf{z}_e^{t-1}|\mathsf{z}_e^0) q_{\varphi,t|s}^{\mathsf{z}_e}(\mathsf{z}_e^t|\mathsf{z}_e^s)\,,$$

where $0\leqslant s\leqslant t$, so that $\tilde{q}_{\varphi,t-1|0,t}^{\mathsf{z}_e}(\mathsf{z}_e^{t-1}|\mathsf{z}_e^t,\mathsf{z}_e^0)$ is a Gaussian probability density function with mean

$$\tilde{\mu}_{\varphi,t-1|0,t}(\mathsf{z}_e^0,\mathsf{z}_e^t) = \frac{\beta_t}{1-\bar{\alpha}_t}\left(z_* + \sqrt{\bar{\alpha}_{t-1}}(\mathsf{z}_e^0 - z_*)\right) + \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\sqrt{\alpha}_t\left(\mathsf{z}_e^t - (1-\sqrt{\alpha_t})z_*\right)$$

and covariance matrix

$$\tilde{\sigma}^2_{\varphi,t-1|0,t} = \frac{\eta^2}{2\vartheta}\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t\,\mathbf{I}_{dN}\,,$$

where $\beta_t = 1-\exp(-2\vartheta\delta_t)$, $\alpha_t = 1-\beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^t \alpha_s$. Note that the bridge sampler proposed in [Ho et al., 2020] is a specific case of this setting with $\eta = \sqrt{2}$, $z_* = 0$ and $\vartheta = 1$.

Choice of denoising model p_θ. Following [Ho et al., 2020], we propose a Gaussian distribution for $p_{\theta,t-1|t}^{\mathsf{z}_e}(\mathsf{z}_e^{t-1}|\mathsf{z}_e^t)$ with mean $\mu_{\theta,t-1|t}(\mathsf{z}_e^t,t)$ and variance $\sigma^2_{\theta,t|t-1}\,\mathbf{I}_{dN}$. In the following, we choose

$$\sigma^2_{\theta,t|t-1} = \frac{\eta^2}{2\vartheta}\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$$

so that the term $\mathcal{L}_t$ of Lemma 3.1 writes

$$2\sigma^2_{\theta,t|t-1}\mathcal{L}_t(\theta,\varphi) = -\mathbb{E}_{q_\varphi}\left[\left\|\mu_{\theta,t-1|t}(\mathsf{z}_e^t,t) - \tilde{\mu}_{\varphi,t-1|0,t}(\mathsf{z}_e^0,\mathsf{z}_e^t)\right\|_2^2\right].$$

In addition, under q_φ, z_e^t has the same distribution as

$$\mathsf{h}_e^t(\mathsf{z}_e^0, \varepsilon_t) = z_* + \sqrt{\bar{\alpha}_t}(\mathsf{z}_e^0 - z_*) + \sqrt{\frac{\eta^2}{2\vartheta}(1-\bar{\alpha}_t)}\varepsilon_t \,,$$

where $\varepsilon_t \sim \mathcal{N}(0, \mathbf{I}_{dN})$. Then, for instance in the case $z_* = 0$, $\tilde{\mu}_{\varphi,t-1|0,t}$ can be reparameterised as follows:

$$\tilde{\mu}_{\varphi,t-1|0,t}(\mathsf{z}_e^0, \mathsf{z}_e^t) = \frac{1}{\sqrt{\alpha_t}}\left(\mathsf{h}_e^t(\mathsf{z}_e^0, \varepsilon_t) - \sqrt{\frac{\eta^2}{2\vartheta(1-\bar{\alpha}_t)}}\beta_t\varepsilon_t\right) .$$

We therefore propose to use

$$\mu_{\theta,t-1|t}(\mathsf{z}_e^t, t) = \frac{1}{\sqrt{\alpha_t}}\left(\mathsf{z}_e^t - \sqrt{\frac{\eta^2}{2\vartheta(1-\bar{\alpha}_t)}}\beta_t\varepsilon_\theta(\mathsf{z}_e^t, t)\right) ,$$

which yields

$$\mathcal{L}_t(\theta, \varphi) = \frac{-\beta_t}{2\alpha_t(1-\bar{\alpha}_{t-1})}\mathbb{E}\left[\left\|\varepsilon_t - \varepsilon_\theta(\mathsf{h}_e^t(\mathsf{z}_e^0, \varepsilon_t), t)\right\|_2^2\right] . \tag{4}$$

Several choices can be proposed to model the function ε_θ. The deep learning architectures considered in the numerical experiments are discussed in Appendix 9 and 10. Similarly to [Ho et al., 2020], we use a stochastic version of our loss function: sample t uniformly in $\{0, \dots, T\}$, and consider $\mathcal{L}_t(\theta, \varphi)$ instead of the full sum over all t. The final training algorithm is described in Algorithm 1 and the sampling procedure in Algorithm 2.

Connections with the VQ-VAE loss function. In the special case where $T = 0$, our loss function can be reduced to a standard VQ-VAE loss function. In that case, write $\mathsf{z}_q = \mathsf{z}_q^0$ and $\mathsf{z}_e = \mathsf{z}_e^0$, the ELBO then becomes:

$$\mathcal{L}(\theta, \varphi) = \mathbb{E}_{q_\varphi}\left[\log p_\theta^x(x|\mathsf{z}_q)\right] + \mathbb{E}_{q_\varphi}\left[\log \frac{p_\theta^{\mathsf{z}_q}(\mathsf{z}_q|\mathsf{z}_e)}{q_\varphi^{\mathsf{z}_q}(\mathsf{z}_q|\mathsf{z}_e)}\right] ,$$

Then, if we assume that $p_\theta^{\mathsf{z}_q}(\mathsf{z}_q|\mathsf{z}_e) = \mathrm{Softmax}\{-\|\mathsf{z}_e - \mathrm{e}_k\|_2^2\}_{1\le k\le K}$ and that $q_\varphi^{\mathsf{z}_q}(\mathsf{z}_q|\mathsf{z}_e)$ is as in [Oord et al., 2017], i.e. a Dirac mass at $\widehat{\mathsf{z}_q} = \mathrm{argmin}_{1\le k\le K}\|\mathsf{z}_e - \mathrm{e}_k\|_2^2$, up to an additive constant, this yields the following random estimation of $\mathbb{E}_{q_\varphi}[\log p_\theta^{\mathsf{z}_q}(\mathsf{z}_q|\mathsf{z}_e)/q_\varphi^{\mathsf{z}_q}(\mathsf{z}_q|\mathsf{z}_e)]$,

$$\widehat{\mathcal{L}}_{\mathsf{z}_q}^{reg}(\theta, \varphi) = \|\mathsf{z}_e - \widehat{\mathsf{z}_q}\|_2 + \log\left(\sum_{k=1}^K \exp\left\{-\|\mathsf{z}_e - \mathrm{e}_k\|_2\right\}\right) .$$

The first term of this loss is the loss proposed in [Oord et al., 2017] which is then split into two parts using the stop gradient operator. The last term is simply the additional normalizing term of $p_\theta^{\mathsf{z}_q}(\mathsf{z}_q|\mathsf{z}_e)$.

Connecting diffusion and discretisation. Similar to the VQ-VAE case above, it is possible to consider only the term $\mathcal{L}_0^{reg}(\theta, \varphi)$ in the case $T > 0$. However, our framework allows for much flexible parameterisation of $p_{\theta,t}^{\mathsf{z}_q}(\mathsf{z}_q^t|\mathsf{z}_e^t)$ and $q_{\varphi,t}^{\mathsf{z}_q}(\mathsf{z}_q^t|\mathsf{z}_e^t)$. For instance, the Gumbel-Softmax trick provides an efficient and differentiable parameterisation. A sample $\mathsf{z}_q^t \sim p_{\theta,t}^{\mathsf{z}_q}(\mathsf{z}_q^t|\mathsf{z}_e^t)$ (resp. $\mathsf{z}_q^t \sim q_{\varphi,t}^{\mathsf{z}_q}(\mathsf{z}_q^t|\mathsf{z}_e^t)$) can be obtained by sampling with probabilities proportional to $\{\exp\{(-\|\mathsf{z}_e - \mathrm{e}_k\|_2^2 + G_k)/\tau_t\}\}_{1\le k\le K}$ (resp. $\{\exp\{(-\|\mathsf{z}_e - \mathrm{e}_k\|_2^2 + \tilde{G}_k)/\tau\}\}_{1\le k\le K}$), where $\{(G_k, \tilde{G}_k)\}_{1\le k\le K}$ are i.i.d. with distribution $\mathrm{Gumbel}(0,1)$, $\tau > 0$, and $\{\tau_t\}_{0\le t\le T}$ are positive time-dependent scaling parameters. In practice, the third part of the objective function can be

repeat
Compute $\mathsf{z}_e^0 = f_\varphi(x)$
Sample $\hat{\mathsf{z}_q^0} \sim q_\varphi(\mathsf{z}_q^0|\mathsf{z}_e^0)$
Compute $\hat{\mathcal{L}}^{rec}(\theta,\varphi) = \log p_\theta^x(x|\hat{\mathsf{z}_q^0})$
Sample $t \sim Uniform(\{0,\dots,T\})$
Sample $\varepsilon_t \sim \mathcal{N}(0, \mathbf{I}_{dN})$
Sample $\mathsf{z}_e^t \sim q_{\varphi,t}(\mathsf{z}_e^t|\mathsf{z}_e^0)$ (using ε_t)
Compute $\hat{\mathcal{L}}_t(\theta,\varphi)$ from $\varepsilon_\theta(\mathsf{z}_e^t, t)$ and ε_t using (4)
Compute $\hat{\mathcal{L}}_t^{reg}(\theta,\varphi)$ from z_e^t *(see text)*
$\hat{\mathcal{L}}(\theta,\varphi) = \hat{\mathcal{L}}^{rec}(\theta,\varphi) + \hat{\mathcal{L}}_t(\theta,\varphi) + \hat{\mathcal{L}}_t^{reg}(\theta,\varphi)$
Perform SGD step on $-\hat{\mathcal{L}}(\theta,\varphi)$
until convergence

Algorithm 1: Training procedure

Sample $\mathsf{z}_e^T \sim \mathcal{N}(0, (2\vartheta)^{-1}\eta^2\mathbf{I}_{dN})$
for $t = T$ **to** 1 **do**
Set $\mathsf{z}_e^{t-1} = \alpha_t^{-1/2}\left(\mathsf{z}_e^t - \sqrt{\frac{\eta^2}{2\vartheta(1-\bar{\alpha}_t)}}\beta_t\varepsilon_\theta(\mathsf{z}_e^t, t)\right)$
end for
Sample $\mathsf{z}_q^0 \sim p_{\theta,0}^{\mathsf{z}_q}(\mathsf{z}_q^0|\mathsf{z}_e^0)$ {*quantisation*}
Sample $x \sim p_\theta^x(x|\mathsf{z}_q^0)$ {*decoder*}

Algorithm 2: Sampling procedure (for $z_* = 0$)

computed efficiently, by using a stochastic version of the ELBO, computing a single $\mathcal{L}_t^{reg}(\theta,\varphi)$ instead of the sum (we use the same t for both parts of the ELBO). The term reduces to:

$$\mathcal{L}_t^{reg}(\theta,\varphi) = -\mathrm{KL}(q_\varphi(\mathsf{z}_q^t|\mathsf{z}_e^t)\|p_\theta(\mathsf{z}_q^t|\mathsf{z}_e^t))\,. \tag{5}$$

This terms connects the diffusion and quantisation parts as it creates a gradient pathway through a step t of the diffusion process, acting as a regularisation on the codebooks and z_e^t. Intuitively, maximizing $\mathcal{L}_t^{reg}(\theta,\varphi)$ accounts for pushing codebooks and z_e^t together or apart depending on the choice of τ, τ_t. The final end-to-end training algorithm is described in Algorithm 1, and further considerations are provided in Appendix 8.

4 Experiments

4.1 Toy Experiment

In order to understand the proposed denoising procedure for VQ-VAE, consider a simple toy setting in which there is no encoder nor decoder, and the codebooks $\{\mathrm{e}_j\}_{0\leqslant j\leqslant K-1}$ are fixed. In this case, with $d = 2$ and $N = 5$, $x = \mathsf{z}_e^0 \in \mathbb{R}^{2\times 5}$. We choose $K = 8$ and the codebooks $\mathrm{e}_j = \mu_j \in \mathbb{R}^2$, $0 \leqslant j \leqslant K-1$, are fixed centers at regular angular intervals in $\mathbb{R}^2$ and shown in Figure 2; the latent states $(\mathsf{z}_q^t)_{1\leq t\leq T}$ lie in $\{\mathrm{e}_0,\dots,\mathrm{e}_7\}^5$. Data generation proceeds as follows. First, sample a sequence of $(q_1,\dots,q_5)$ in $\{0,\dots,7\}$: q_1 has a uniform distribution, and, for $s \in \{0,1,2,3\}$, $q_{s+1} = q_s + b_s \mod 8$, where b_s are independent Bernoulli samples with parameter $1/2$ taking values in $\{-1,1\}$. Conditionally on $(q_1,\dots,q_5)$, x is a Gaussian random vector with mean $(\mathrm{e}_{q_1},\dots,\mathrm{e}_{q_5})$ and variance $\mathbf{I}_{2\times 5}$.

We train our bridge procedure with $T = 50$ timesteps, $\vartheta = 2, \eta = 0.1$, other architecture details and the neural network $\varepsilon_\theta(\mathsf{z}_e^t, t)$ are described in Appendix 10. Forward noise process and denoising using $\varepsilon_\theta(\mathsf{z}_e^t, t)$

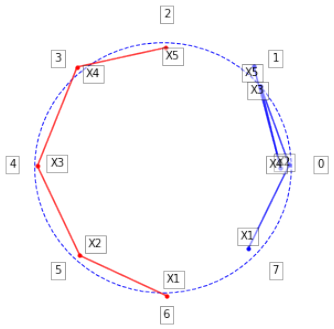

Figure 2: Toy dataset, with $K = 8$ centroids, and two samples $x = (x_1, x_2, x_3, x_4, x_5)$ in $\mathbb{R}^{2\times 5}$ each displayed as 5 points in $\mathbb{R}^2$ (blue and red points), corresponding to the discrete sequences (red) $(6, 5, 4, 3, 2)$ and (blue) $(7, 0, 1, 0, 1)$.

are showcased in Figure 3, and more illustrations and experiments can be found in Appendix 10.

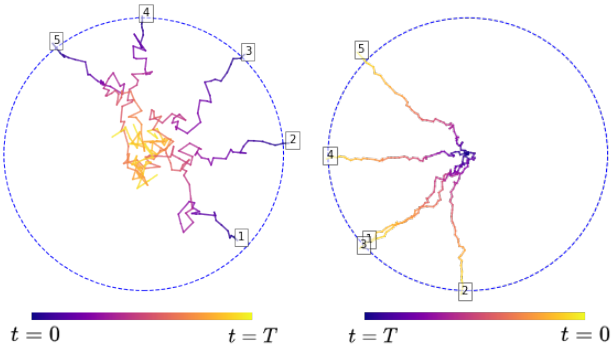

Figure 3: (Left) Forward noise process for one sample. First, one data is drawn ($\mathsf{z}_e^0(x) = x$ in the toy example) and then $\{\mathsf{z}_e^t\}_{1\leq t\leq T}$ are sampled under q_φ and displayed. (Right) Reverse process for one sample $\mathsf{z}_e^T \sim \mathcal{N}(0, (2\vartheta)^{-1}\eta^2\mathbf{I}_{dN})$. As expected, the last sample z_e^0 reaches the neighborhood of 5 codebooks.

End-to-end training. Contrary to VQ-VAE procedures in which the encoder, decoder and codebooks are trained separately from the prior, we can train the bridge prior alongside the codebooks. Consider a new setup, in which the $K = 8$ codebooks are randomly initialized and considered as parameters of our model (they are no longer fixed to the centers of the data generation process μ_j). The first part of our loss function, in conjunction with the Gumbel-Softmax trick makes it possible to train all the parameters of the model end-to-end. Details of the procedure and results are shown in Appendix 10.

4.2 Image Synthesis

In this section, we focus on image synthesis using CIFAR10 and miniImageNet datasets. The goal is to evaluate the efficiency and properties of our model compared to the original PixelCNN. Note that for fair comparisons, the encoder, decoder and codebooks are pretrained and fixed for all models, only the prior is trained and evaluated here. As our goal is the comparison of priors, we did not focus on building the most efficient VQ-VAE, but rather a reasonable model in terms of size and efficiency.

CIFAR10. The CIFAR dataset consists of inputs x of dimensions 32×32 with 3 channels. The encoder projects the input into a grid of continuous values z_e^0 of dimension $8 \times 8 \times 128$. After discretisation, $\{\mathsf{z}_q^t\}_{0 \leqslant t \leqslant T}$ are in a discrete latent space induced by the VQ-VAE which consists of values in $\{1, \dots, K\}^{8 \times 8}$ with $K = 256$. The pre-trained VQ-VAE reconstructions can be seen in Figure 13 in Appendix 11.

miniImageNet. *mini*ImageNet was introduced by [Vinyals et al., 2016] to offer more complexity than CIFAR10, while still fitting in memory of modern machines. 600 images were sampled for 100 different classes from the original ImageNet dataset, then scaled down, to obtain 60,000 images of dimension 84×84. In our experiments, we trained a VQVAE model to project those input images into a grid of continuous values z_e^0 of dimensions $21 \times 21 \times 32$, see Figure 15 in Appendix 11. The associated codebook contains $K = 128$ vectors of dimension 32.

Prior models. Once the VQ-VAE is trained on the miniImageNet and CIFAR datasets, the $84 \times 84 \times 3$ and $32 \times 32 \times 3$ images respectively are passed to the encoder and result in 21×21 and 8×8 feature maps respectively. From this model, we extract the discrete latent states from training samples to train a PixelCNN prior and the continuous latent states for our diffusion. Concerning our diffusion prior, we choose the Ornstein-Uhlenbeck process setting $\eta = \sqrt{2}$, $z_* = 0$ and $\vartheta = 1$, with $T = 1000$.

End-to-End Training. As an additional experiment, we propose an End-to-End training of the VQ-VAE and the diffusion process. To speed up training, we first start by pretraining the VQ-VAE, then learn the parameters of our diffusion prior alongside all the VQ-VAE parameters (encoder, decoder and codebooks). Note that in this setup, we cannot directly compare the NLL to PixelCNN or our previous diffusion model as the VQ-VAE has changed, but we can compare image generation metrics such as FID and sample quality.

4.3 Quantitative results

We benchmarked our model using three metrics, in order to highlight the performances of the proposed prior, the quality of produced samples as well as the associated computation costs. Results are given as a comparison to the original PixelCNN prior for both the *mini*ImageNet (see Table 2) and the CIFAR10 (see Table 3) datasets.

Negative Log Likelihood. Unlike most related papers, we are interested in computing the Negative Log Likelihood (NLL) directly in the latent space, as to evaluate the capacity of the priors to generate coherent latent maps. To this end, we mask a patch of the original latent space, and reconstruct the missing part, similar to image inpainting, following for instance [Van Oord et al., 2016]. In the case of our prior, for each sample x, we mask an area of the continuous latent state z_e^0, i.e. we mask some components of z_e^0, and aim at sampling the missing components given the observed ones using the prior model. Let $\underline{\mathsf{z}_q}^0$ and $\underline{\mathsf{z}_e}^0$ (resp. $\overline{\mathsf{z}_q}^0$ and $\overline{\mathsf{z}_e}^0$) be the masked (resp. observed) discrete and continuous latent variables. The target conditional likelihood is

$$\begin{aligned} p_\theta(\underline{\mathsf{z}_q}^0 | \overline{\mathsf{z}_e}^0) &= \int p_\theta(\underline{\mathsf{z}_q}^0, \underline{\mathsf{z}_e}^0 | \overline{\mathsf{z}_e}^0) \mathrm{d}\underline{\mathsf{z}_e}^0 \,, \\ &= \int p_\theta(\underline{\mathsf{z}_q}^0 | \underline{\mathsf{z}_e}^0) p_\theta(\underline{\mathsf{z}_e}^0 | \overline{\mathsf{z}_e}^0) \mathrm{d}\underline{\mathsf{z}_e}^0 \,. \end{aligned}$$

This likelihood is intractable and replaced by a simple Monte Carlo estimate $\hat{p}_\theta(\underline{\mathsf{z}_q}^0 | \underline{\mathsf{z}_e}^0)$ where $\underline{\mathsf{z}_e}^0 \sim p_\theta(\underline{\mathsf{z}_e}^0 | \overline{\mathsf{z}_e}^0)$. Note that conditionally on $\underline{\mathsf{z}_e}^0$ the components of $\underline{\mathsf{z}_q}^0$ are assumed to be independent but $\underline{\mathsf{z}_e}^0$ are

sampled jointly under $p_\theta(\underline{\mathsf{z}_e}^0|\overline{\mathsf{z}_e}^0)$. As there are no continuous latent data in PixelCNN, $p_\theta(\underline{\mathsf{z}_q}^0|\overline{\mathsf{z}_q}^0)$ can be directly evaluated.

Fréchet Inception Distance. We report Fréchet Inception Distance (FID) scores by sampling a latent discrete state $\mathsf{z}_q \in \mathcal{E}^N$ from the prior, and computing the associated image through the VQ-VAE decoder. In order to evaluate each prior independently from the encoder and decoder networks, these samples are compared to VQ-VAE reconstructions of the dataset images.

Kullback-Leibler divergence. In this experiment, we draw $M = 1000$ samples from test set and encode them using the trained VQ-VAE, and then draw as many samples from the pixelCNN prior, and our diffusion prior. We propose then to compute the empirical Kullback Leibler (KL) divergence between original and sampled distribution at each pixel. Figure 4 highlights that PixelCNN performs poorly on the latest pixels (at the bottom) while our method remains consistent. This is explained by our denoising process in the continuous space which uses all pixels jointly while PixelCNN is based on an autoregressive model.

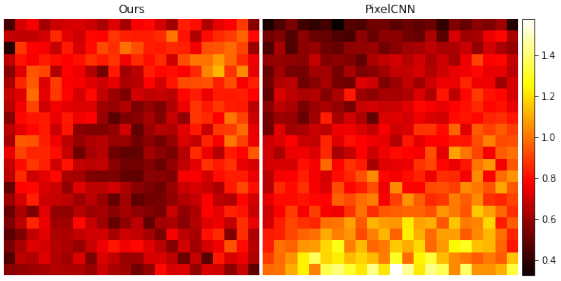

Figure 4: KL Distance between the true empirical distribution and both prior distributions in the latent space. Darker squares indicates lower (better) values.

	KL
Ours	**0.713**
PixelCNN	0.809

Table 1: Averaged KL metric on the feature map.

Computation times. We evaluated the computation cost of sampling a batch of 32 images, on a GTX TITAN Xp GPU card. Note that the computational bottleneck of our model consists of the $T = 1000$ sequential diffusion steps (rather than the encoder/decoder which are very fast in comparison). Therefore, a diffusion speeding technique such as the one described in [Song et al., 2021] would be straightforward to apply and would likely provide a $\times 50$ speedup as mentioned in the paper.

4.4 Qualitative results

Sampling from the prior. Samples from the PixelCNN prior are shown in Figure 5b and samples from our prior in Figure 5a. Additional samples are given in Appendix 11. Note that contrary to original VQ-VAE prior, the prior is not conditioned on a class, which makes the generation less specific and more difficult. However, the produced samples illustrate that our prior can generate a wide variety of images which show a large-scale spatial coherence in comparison with samples from PixelCNN.

Table 2: Results on *mini*ImageNet. Metrics are computed on the validation dataset. The means are displayed along with the standard deviation in parenthesis.

	NLL	FID	s/sample
PixelCNN [Oord et al., 2017]	1.00 ($\pm$0.05)	98	10.6s ($\pm 28ms$)
Ours	0.94 ($\pm$0.02)	99	1.7s ($\pm 10ms$)

Table 3: Results on CIFAR10. Metrics are computed on the validation dataset. The means are displayed along with the standard deviation in parenthesis.

	NLL	FID	s/sample
PixelCNN [Oord et al., 2017]	1.41 ($\pm$0.06)	109	0.21 ($\pm 0.8ms$)
Ours	1.33 ($\pm$0.18)	104	0.05s ($\pm 0.5ms$)
Ours end-to-end	1.59 ($\pm$0.27)[1]	92	0.11s ($\pm 0.5ms$)

(a) Samples from our diffusion prior.

(b) Samples from the PixelCNN prior.

Figure 5: Comparison between samples from our diffusion-based prior (top) and PixelCNN prior (bottom).

Conditional sampling. As explained in Section 4.3, for each sample x, we mask some components of $\mathsf{z}_e^0(x)$, and aim at sampling the missing components given the observed ones using the prior models. This conditional denoising process is further explained for our model in Appendix 7. To illustrate this setting, we show different conditional samples for 3 images in Figure 6 and Figure 7 for both the PixelCNN prior and ours. In Figure 6, the mask corresponds to a 9×9 centered square over the 21×21 feature map. In Figure 7, the mask corresponds to a 9×9 top left square. These figures illustrate that our diffusion model is

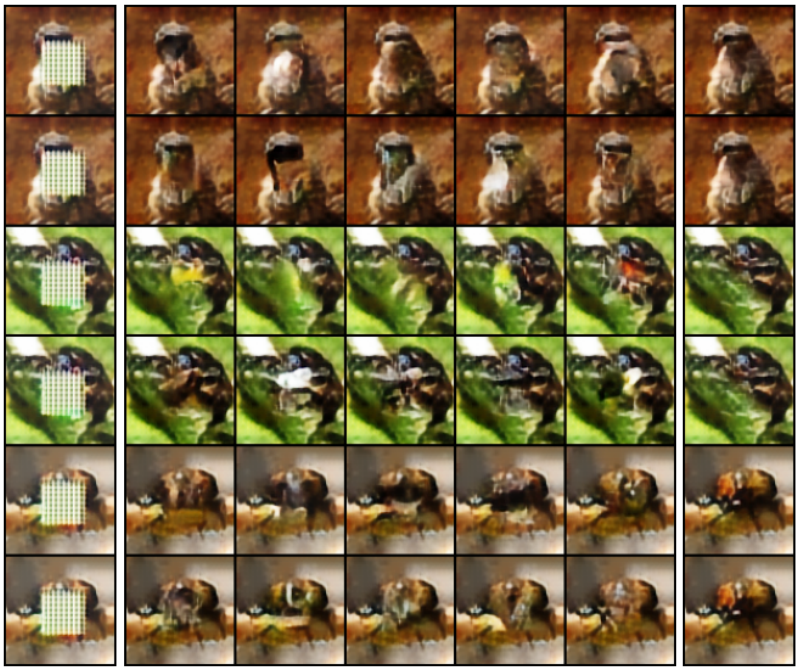

Figure 6: Conditional sampling with centered mask: for each of the 3 different images, samples from our diffusion are on top and from PixelCNN on the bottom. For each row: the image on the left is the VQVAE masked reconstruction, the image on the right is the full VQ-VAE reconstruction. Images in-between are independent conditional samples from the models.

much less sensitive to the selected masked region than PixelCNN. This may be explained by the use of our denoising function ε_θ which depends on all conditioning pixels while PixelCNN uses a hierarchy of masked convolutions to enforce a specific conditioning order. Additional conditional sampling experiments are given in Appendix 11.

Denoising chain. In addition to the conditional samples, Figure 8 shows the conditional denoising process at regularly spaced intervals, and Figure 9 shows unconditional denoising. Each image of the chain is generated by passing the predicted $\mathbf{z}_q^t$ through the VQ-VAE decoder.

5 Conclusion

This work introduces a new mathematical framework for VQ-VAEs which includes a diffusion probabilistic model to learn the dependencies between the continuous latent variables alongside the encoding and decoding part of the model. We showed conceptual improvements of our model over the VQ-VAE prior, as well as first numerical results on middle scale image generation. We believe that these first numerical experiments open up many research avenues: scaling to larger models, optimal scaling of the hyperparameters, including standard tricks from other diffusion methods, studying the influence of regulazation loss for end-to-end training, etc. We hope that this framework will serve as a sound and stable foundation to derive future generative models.

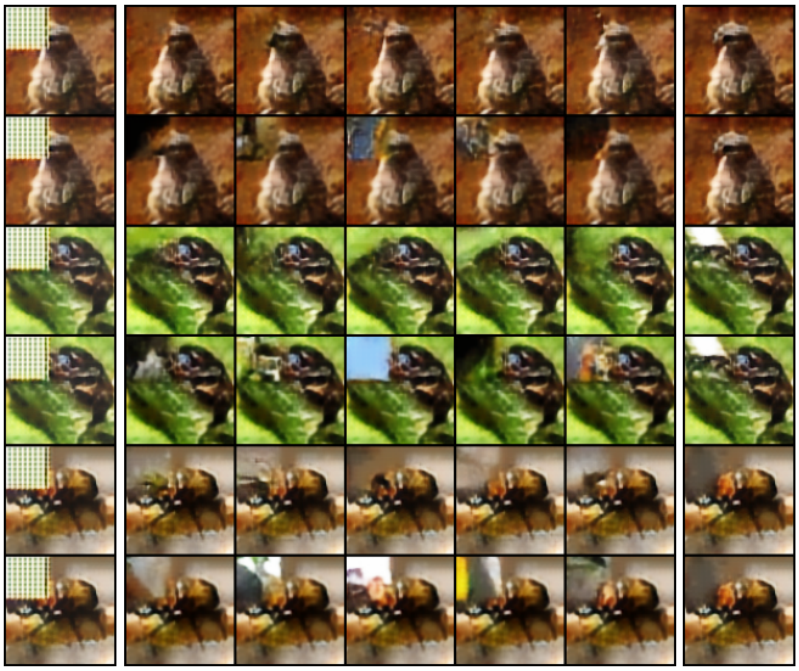

Figure 7: Conditional sampling with top left mask: for each of the 3 different images, samples from our diffusion are on top and from PixelCNN on the bottom. For each row: the image on the left is the VQVAE masked reconstruction, the image on the right is the full VQ-VAE reconstruction. Images in-between are independent conditional samples from the models.

Figure 8: Sampling denoising chain from $t = 500$ up to $t = 0$, shown at regular intervals, conditioned on the outer part of the picture. We show only the last 500 steps of this process, as the first 500 steps are not visually informative. The sampling procedure is described in Appendix 7.

Figure 9: Sampling denoising chain from $t = 500$ up to $t = 0$, shown at regular intervals, unconditional. We show only the last 500 steps of this process, as the first 500 steps are not visually informative. The sampling procedure is described in Algorithm 2

Acknowledgements

The work of Max Cohen was supported by grants from Région Ile-de-France. Charles Ollion and Guillaume Quispe benefited from the support of the Chair "New Gen RetAIl" led by l'X – École Polytechnique and the Fondation de l'École Polytechnique, sponsored by Carrefour.

6 Details on the loss function

Proof of Lemma 3.1. By definition,

$$\mathcal{L}(\theta, \varphi) = \mathbb{E}_{q_\varphi}\left[\log \frac{p_\theta(\mathsf{z}_q^{0:T}, \mathsf{z}_e^{0:T}, x)}{q_\varphi(\mathsf{z}_q^{0:T}, \mathsf{z}_e^{0:T}|x)}\right],$$

which yields

$$\mathcal{L}(\theta, \varphi) = \mathbb{E}_{q_\varphi}\left[\log p_\theta^x(x|\mathsf{z}_q^0)\right] + \mathbb{E}_{q_\varphi}\left[\log \frac{p_\theta^{\mathsf{z}_q}(\mathsf{z}_q^{0:T}|\mathsf{z}_e^{0:T})}{q_\varphi^{\mathsf{z}_q}(\mathsf{z}_q^{0:T}|\mathsf{z}_e^{0:T})}\right] + \mathbb{E}_{q_\varphi}\left[\log \frac{p_\theta^{\mathsf{z}_e}(\mathsf{z}_e^{0:T})}{q_\varphi^{\mathsf{z}_e}(\mathsf{z}_e^{0:T}|x)}\right].$$

The last term may be decomposed as

$$\mathbb{E}_{q_\varphi}\left[\log \frac{p_\theta^{\mathsf{z}_e}(\mathsf{z}_e^{0:T})}{q_\varphi^{\mathsf{z}_e}(\mathsf{z}_e^{0:T}|x)}\right] = \mathbb{E}_{q_\varphi}\left[\log p_{\theta,T}^{\mathsf{z}_e}(\mathsf{z}_e^T)\right] + \sum_{t=1}^{T} \mathbb{E}_{q_\varphi}\left[\log \frac{p_{\theta,t-1|t}^{\mathsf{z}_e}(\mathsf{z}_e^{t-1}|\mathsf{z}_e^t)}{q_{\varphi,t|t-1}^{\mathsf{z}_e}(\mathsf{z}_e^t|\mathsf{z}_e^{t-1})}\right]$$

and

$$\mathbb{E}_{q_\varphi}\left[\log \frac{p_\theta^{\mathsf{z}_e}(\mathsf{z}_e^{0:T})}{q_\varphi^{\mathsf{z}_e}(\mathsf{z}_e^{0:T}|x)}\right] = \mathbb{E}_{q_\varphi}\left[\log p_{\theta,T}^{\mathsf{z}_e}(\mathsf{z}_e^T)\right] + \mathbb{E}_{q_\varphi}\left[\log \frac{p_{\theta,0|1}^{\mathsf{z}_e}(\mathsf{z}_e^0|\mathsf{z}_e^1)}{q_{\varphi,1|0}^{\mathsf{z}_e}(\mathsf{z}_e^1|\mathsf{z}_e^0)}\right] + \sum_{t=2}^{T} \mathbb{E}_{q_\varphi}\left[\log \frac{p_{\theta,t-1|t}^{\mathsf{z}_e}(\mathsf{z}_e^{t-1}|\mathsf{z}_e^t)}{q_{\varphi,t|t-1}^{\mathsf{z}_e}(\mathsf{z}_e^t|\mathsf{z}_e^{t-1})}\right].$$

By (1),

$$\mathbb{E}_{q_\varphi}\left[\log \frac{p_\theta^{\mathsf{z}_e}(\mathsf{z}_e^{0:T})}{q_\varphi^{\mathsf{z}_e}(\mathsf{z}_e^{0:T}|x)}\right] = \mathbb{E}_{q_\varphi}\left[\log \frac{p_{\theta,T}^{\mathsf{z}_e}(\mathsf{z}_e^T)}{q_{\varphi,T|0}^{\mathsf{z}_e}(\mathsf{z}_e^T|\mathsf{z}_e^0)}\right] + \sum_{t=2}^{T} \mathbb{E}_{q_\varphi}\left[\log \frac{p_{\theta,t-1|t}^{\mathsf{z}_e}(\mathsf{z}_e^{t-1}|\mathsf{z}_e^t)}{q_{\varphi,t-1|0,t}^{\mathsf{z}_e}(\mathsf{z}_e^{t-1}|\mathsf{z}_e^0, \mathsf{z}_e^t)}\right] + \mathbb{E}_{q_\varphi}\left[\log p_{\theta,0|1}^{\mathsf{z}_e}(\mathsf{z}_e^0|\mathsf{z}_e^1)\right],$$

which concludes the proof. □

7 Inpainting diffusion sampling

We consider the case in which we know a sub-part of the picture $\overline{X}$, and want to predict the complementary pixels $\underline{X}$. Knowing the corresponding n latent vectors $\overline{\mathsf{z}_e}^0$ which result from $\underline{X}$ through the encoder, we sample $N-n$ $\underline{\mathsf{z}_e}^T$ from the uninformative distribution $\underline{\mathsf{z}_e}^T \sim \mathcal{N}(0, (2\vartheta)^{-1}\eta^2 \mathbf{I}_{d\times(N-n)})$. In order to produce the chain of samples z_e^{t-1} from z_e^t we then follow the following procedure.

- $\underline{\mathsf{z}_e}^{t-1}$ is predicted from z_e^t using the neural network predictor, similar to the unconditioned case.
- Sample $\overline{\mathsf{z}_e}^{t-1}$ using the forward bridge noising process.

8 Additional regularisation considerations

We consider here details about the parameterisation of $p_\theta^{\mathsf{z}_q}(\mathsf{z}_q^t|\mathsf{z}_e^t)$ and $q_\varphi^{\mathsf{z}_q}(\mathsf{z}_q^t|\mathsf{z}_e^t)$ in order to compute $\mathcal{L}_t^{reg}(\theta,\varphi)$. Using the Gumbel-Softmax formulation provides an efficient and differentiable parameterisation.

$$p_{\theta,t}^{\mathsf{z}_q}(\mathsf{z}_q^t = \cdot|\mathsf{z}_e^t) = \mathrm{Softmax}\{(-\|\mathsf{z}_e - \mathrm{e}_k\|_2^2 + G_k)/\tau_t\}_{1\leqslant k\leqslant K}\,,$$
$$q_{\varphi,t}(\mathsf{z}_q^t = \cdot|\mathsf{z}_e^t) = \mathrm{Softmax}\{(-\|\mathsf{z}_e - \mathrm{e}_k\|_2^2 + \tilde{G}_k)/\tau\}_{1\leqslant k\leqslant K}\,,$$

where $\{(G_k, \tilde{G}_k)\}_{1\leqslant k\leqslant K}$ are i.i.d. with distribution $\mathrm{Gumbel}(0,1)$, $\tau > 0$, and $\{\tau_t\}_{0\leqslant t\leqslant T}$ are positive time-dependent scaling parameters. Then, up to the additive normalizing terms,

$$\mathcal{L}_t^{reg}(\theta,\varphi) = \mathbb{E}_{q_\varphi}\left[\log \frac{p_{\theta,t}^{\mathsf{z}_q}(\mathsf{z}_q^t|\mathsf{z}_e^t)}{q_{\varphi,t}^{\mathsf{z}_q}(\mathsf{z}_q^t|\mathsf{z}_e^t)}\right] = \left(-\frac{1}{\tau_t} + \frac{1}{\tau}\right)\|\mathsf{z}_e^t - \widehat{\mathsf{z}_q^t}\|_2^2 - \frac{\tilde{G}_k}{\tau} + \frac{G_k}{\tau_t}\,,$$

where $\widehat{\mathsf{z}_q^t} \sim q_{\varphi,t}^{\mathsf{z}_q}(\mathsf{z}_q^t|\mathsf{z}_e^t)$. Considering only the first term which depend on z_e^t and produce non-zero gradients, we get:

$$\mathcal{L}_t^{reg}(\theta,\varphi) = \gamma_t\|\mathsf{z}_e^t - \widehat{\mathsf{z}_q^t}\|_2^2$$

where $\gamma_t = -1/\tau_t + 1/\tau$ drives the behavior of the regulariser. By choosing is γ_t negative for large t, the regulariser pushes the codebooks away from z_e^t, which prevents too early specialization, or matching of codebooks with noise, as $\mathsf{z}_e^{t\approx T}$ is close to the uninformative distribution. Finally, for small t, choosing γ_t positive helps matching codebooks with z_e when the corruption is small. In practice $\tau = 1$ and a simple schedule from 10 to 0.1 for τ_t was considered in this work.

9 Neural Networks

For $\varepsilon_\theta(\mathsf{z}_e^t, t)$, we use a U-net like architecture similar to the one mentioned in [Ho et al., 2020]. It consists of a deep convolutional neural network with 57M parameters, which is slightly below the PixelCNN architecture (95.8M parameters). The VQ-VAE encoder / decoders are also deep convolutional networks totalling 65M parameters.

10 Toy Example Appendix

Parameterisation We consider a neural network to model $\varepsilon_\theta(\mathsf{z}_e^t, t)$. The network shown in Figure 10 consists of a time embedding similar to [Ho et al., 2020], as well as a few linear or 1D-convolutional layers, totalling around 5000 parameters.

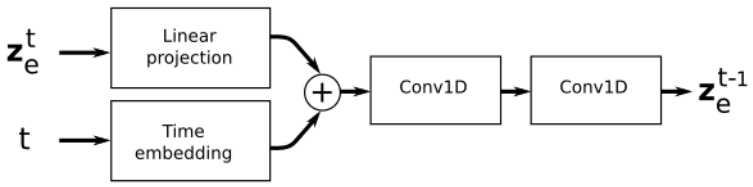

Figure 10: Graphical representation of the neural network used for the toy dataset.

For the parameterisation of the quantization part, we choose $p^{\mathsf{z}_q}_{\theta,t}(\mathsf{z}_q^t = \mathrm{e}_j|\mathsf{z}_e^t) = \mathrm{Softmax}_{1\leq k\leq K}\{-\|\mathsf{z}_e - \mathrm{e}_k\|_2\}_j$, and the same parameterisation for $q^{\mathsf{z}_q}_{\varphi,t}(\mathsf{z}_q^t|\mathsf{z}_e^t)$. Therefore our loss simplifies to:

$$\mathcal{L}(\theta,\varphi) = \mathbb{E}_{q_\varphi}\left[\log p^x_\theta(x|\mathsf{z}_q^0)\right] + \mathcal{L}_t(\theta,\varphi)\,,$$

where t is sampled uniformly in $\{0,\dots,T\}$.

t	NN sequence
50	(0, 7, 3, 6, 2)
40	(6, 5, 5, 5, 3)
30	(5, 5, 5, 4, 2)
20	(6, 6, 5, 4, 3)
10	(5, 6, 5, 4, 3)
0	(5, 6, 5, 4, 3)

Table 4: Discrete samples during diffusion process. The discrete sequence is obtained by computing the nearest neighbour centroid μ_j for each X_s^t. At $t = 0$, X^0 is sampled from a centered Gaussian distribution with small covariance matrix $(2\vartheta)^{-1}\eta^2\mathbf{I}_{2\times 5}$, resulting in a uniform discrete sequence, as all centroids have a similar unit norm.

Discrete samples during diffusion process Discrete sequences corresponding to the denoising diffusion process shown in Figure 3 are shown in Table 4.

End-to-end training In order to train the codebooks alongside the diffusion process, we need to backpropagate the gradient of the likelihood of the data z_e given a z_e^0 reconstructed by the diffusion process (corresponding to $\mathcal{L}^{rec}(\theta,\varphi)$). We use the Gumbel-Softmax parameterisation in order to obtain a differentiable process and update the codebooks e_j.

In this toy example, the use of the third part of the loss $\sum_{t=0}^T \mathcal{L}_t^{reg}(\theta,\varphi)$ is not mandatory as we obtain good results with $\mathcal{L}_t^{reg}(\theta,\varphi) = 0$, which means parametrising $p^{\mathsf{z}_q}_{\theta,t}(\mathsf{z}_q^t|\mathsf{z}_e^t) = q^{\mathsf{z}_q}_{\varphi,t}(\mathsf{z}_q^t|\mathsf{z}_e^t)$. However we noticed that $\mathcal{L}_t^{reg}(\theta,\varphi)$ is useful to improve the learning of the codebooks. If we choose γ_t to be decreasing with time t, we have the following. When t is low, the denoising process is almost over, $\mathcal{L}_t^{reg}(\theta,\varphi)$ pushes z_e and the selected z_q to close together: $\|\mathsf{z}_e\| \sim 1$, then $\|\mathsf{z}_e^t\|$ will be likely near a specific e_j and far from the others; therefore only a single codebook is selected and receives gradient. When t is high, $\|\mathsf{z}_e^t\| \sim 0$ and the Gumbel-Softmax makes it so that all codebooks are equidistant from $\|\mathsf{z}_e^t\|$ and receive non-zero gradient. This naturally solves training problem associated with dead codebooks in VQ-VAEs. Joint training of the denoising and codebooks yield excellent codebook positionning as shown in Figure 11.

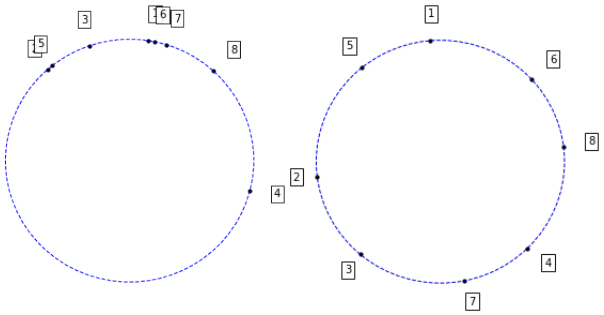

Figure 11: Left, initial random codebooks positions. Right, after training, position of codebook vectors. Note that the codebook indexes do not match the indexes of the Gaussians, the model learnt to make the associations between neighboring centroids in a different order.

Toy Diffusion inpainting We consider a case in which we want to reconstruct an x while we only know one (or a few) dimensions, and sample the others. Consider that x is generated using a sequence $q = (q_1, q_2, q_", q_4, q_5)$ where the last one if fixed $q_1 = 0, q_5 = 4$. Then, knowing q_1, q_5, we sample q_2, q_3, q_4, as shown in Figure 12.

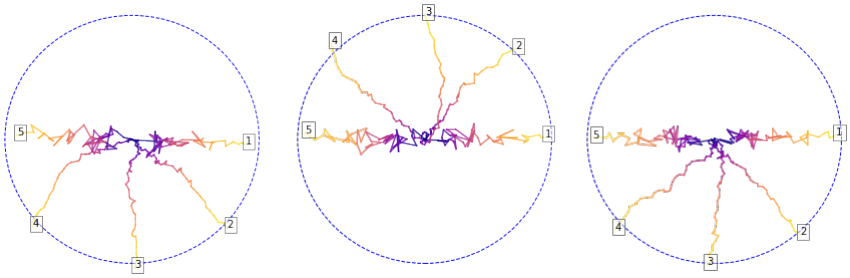

Figure 12: Three independent sampling of X using a trained diffusion bridge, with fixed $q_1 = 0, q_5 = 4$. The three corresponding sequences are $(0, 7, 6, 5, 4)$, $(0, 1, 2, 3, 4)$, $(0, 7, 6, 5, 4)$ all valid sequences.

11 Additional visuals

11.1 Cifar

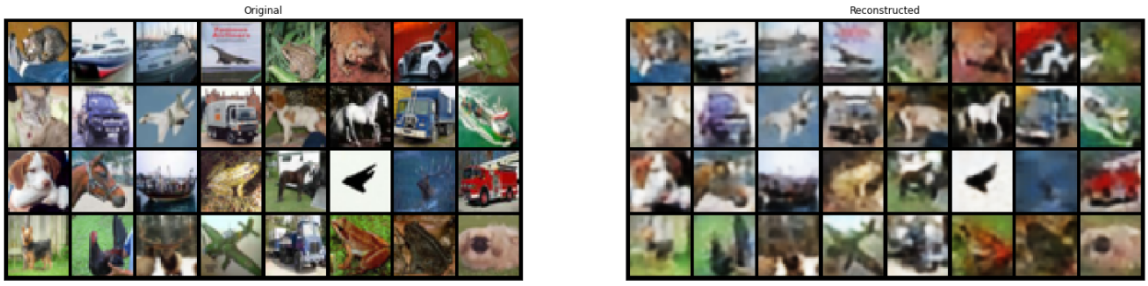

Figure 13: Reconstruction of the VQVAE model used in the following benchmarks.

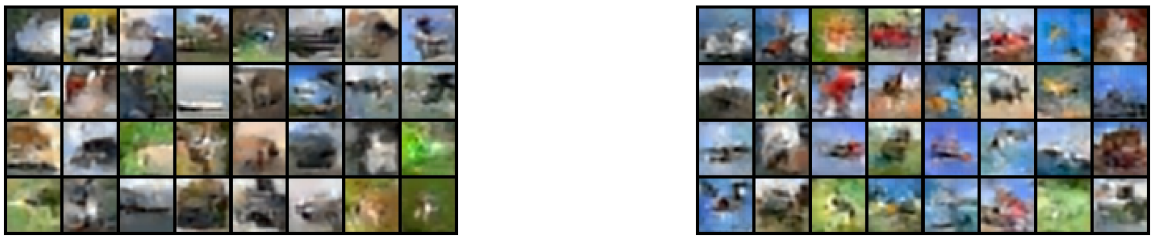

Figure 14: Samples from the PixelCNN prior (left) and from our diffusion prior (right) on CIFAR10.

11.2 MiniImageNet

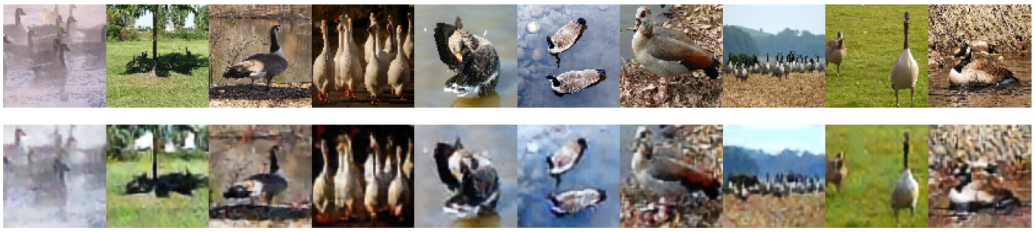

Figure 15: Reconstruction of the trained VQ-VAE on the *mini*ImageNet dataset. Original images are encoded, discretised, and decoded.

Figure 16: Samples from our model for the miniimagenet dataset

Figure 17: Conditional sampling: Top: reconstructions from the vqvae of originals images, Middle: conditional sampling with the left side of the image as condition, for our model. Bottom 1 and 2: conditional sampling in the same context with the PixelCNN prior.

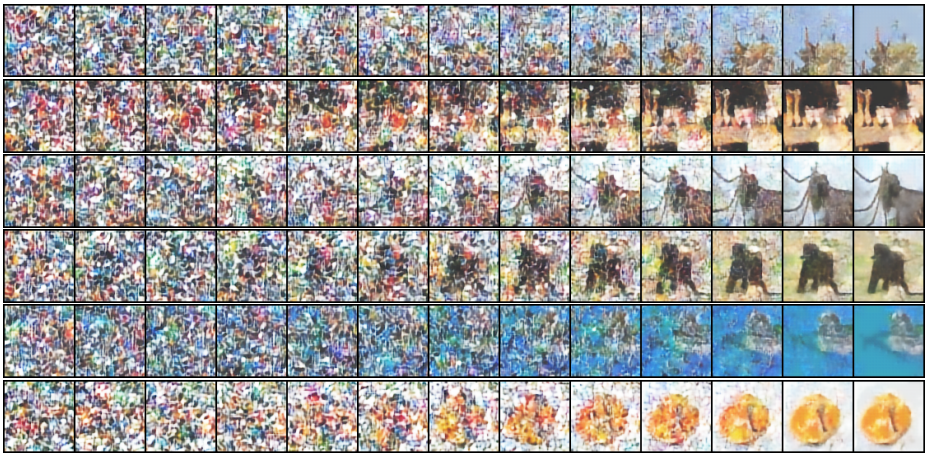

Figure 18: Sampling denoising chain from up to $t = 0$, shown at regular intervals, conditioned on the left part of the picture. The sampling procedure is described in Appendix 7.

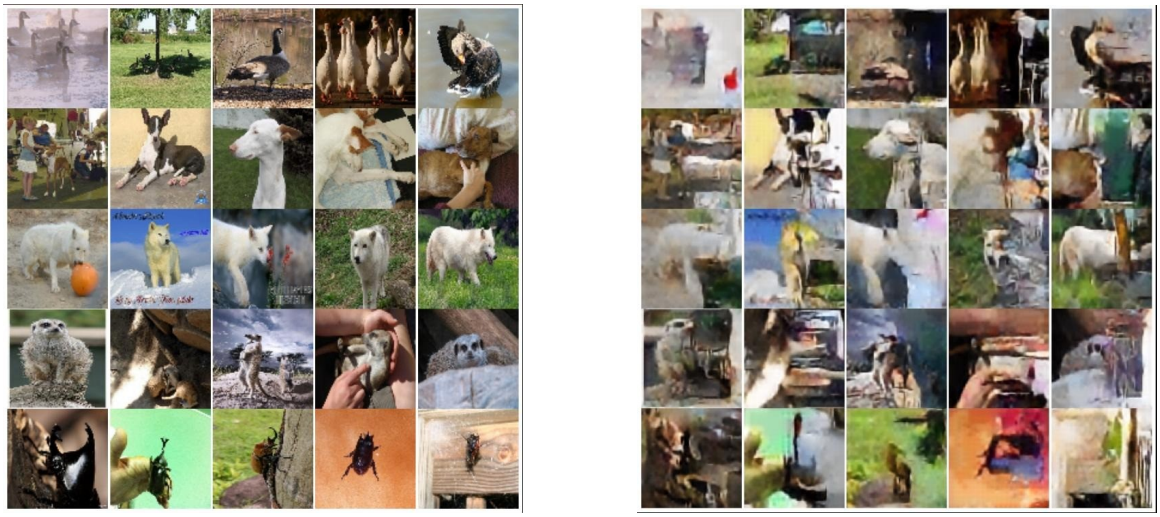

Figure 19: Conditional sampling with the PixelCNN prior. **Left**: original images, **Right**: conditional sampling with the left side of the image as condition. Each row represents a class of the validation set of the *mini*ImageNet dataset.

References

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

[Beskos et al., 2008] Beskos, A., Roberts, G., Stuart, A., and Voss, J. (2008). Mcmc methods for diffusion bridges. *Stochastics and Dynamics*, 8(03):319–350.

[Bladt et al., 2016] Bladt, M., Finch, S., and Sørensen, M. (2016). Simulation of multivariate diffusion bridges. *Journal of the Royal Statistical Society: Series B: Statistical Methodology*, pages 343–369.

[Chen et al., 2018] Chen, X., Mishra, N., Rohaninejad, M., and Abbeel, P. (2018). Pixelsnail: An improved autoregressive generative model. In *International Conference on Machine Learning*, pages 864–872. PMLR.

[De Bortoli et al., 2021] De Bortoli, V., Doucet, A., Heng, J., and Thornton, J. (2021). Simulating diffusion bridges with score matching. *arXiv preprint arXiv:2111.07243*.

[Esser et al., 2021] Esser, P., Rombach, R., and Ommer, B. (2021). Taming transformers for high-resolution image synthesis. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12873–12883.

[Gu et al., 2021] Gu, S., Chen, D., Bao, J., Wen, F., Zhang, B., Chen, D., Yuan, L., and Guo, B. (2021). Vector quantized diffusion model for text-to-image synthesis. *arXiv preprint*.

[Ho et al., 2020] Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems (NeurIPS 2021)*, 34.

[Hoogeboom et al., 2021] Hoogeboom, E., Nielsen, D., Jaini, P., Forré, P., and Welling, M. (2021). Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in Neural Information Processing Systems (NeurIPS 2021)*, 34.

[Lin et al., 2010] Lin, M., Chen, R., and Mykland, P. (2010). On generating monte carlo samples of continuous diffusion bridges. *Journal of the American Statistical Association*, 105(490):820–838.

[Mittal et al., 2021] Mittal, G., Engel, J., Hawthorne, C., and Simon, I. (2021). Symbolic music generation with diffusion models. *arXiv preprint arXiv:2103.16091*.

[Oord et al., 2016] Oord, A. v. d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. (2016). Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*.

[Oord et al., 2017] Oord, A. v. d., Vinyals, O., and Kavukcuoglu, K. (2017). Neural discrete representation learning. *Advances in neural information processing systems (NeurIPS 2017)*.

[Ramesh et al., 2021] Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. (2021). Zero-shot text-to-image generation. 139:8821–8831.

[Razavi et al., 2019] Razavi, A., van den Oord, A., and Vinyals, O. (2019). Generating diverse high-fidelity images with vq-vae-2. In *Advances in neural information processing systems (NeurIPS 2019)*, pages 14866–14876.

[Salimans et al., 2017] Salimans, T., Karpathy, A., Chen, X., and Kingma, D. P. (2017). Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications.

[Sohl-Dickstein et al., 2015] Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. 37:2256–2265.

[Song et al., 2021] Song, J., Meng, C., and Ermon, S. (2021). Denoising diffusion implicit models.

[Song and Ermon, 2019] Song, Y. and Ermon, S. (2019). Generative modeling by estimating gradients of the data distribution. 32.

[Vahdat et al., 2021] Vahdat, A., Kreis, K., and Kautz, J. (2021). Score-based generative modeling in latent space.

[van den Oord et al., 2016] van den Oord, A., Kalchbrenner, N., Vinyals, O., Espeholt, L., Graves, A., and Kavukcuoglu, K. (2016). Conditional image generation with pixelcnn decoders.

[Van Oord et al., 2016] Van Oord, A., Kalchbrenner, N., and Kavukcuoglu, K. (2016). Pixel recurrent neural networks. In *International Conference on Machine Learning*, pages 1747–1756. PMLR.

[Vinyals et al., 2016] Vinyals, O., Blundell, C., Lillicrap, T., kavukcuoglu, k., and Wierstra, D. (2016). Matching networks for one shot learning. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

[Willetts et al., 2021] Willetts, M., Miscouridou, X., Roberts, S., and Holmes, C. (2021). Relaxed-responsibility hierarchical discrete VAEs. *ArXiv:2007.07307*.