Structured Denoising Diffusion Models in Discrete State-Spaces

Anonymous Author(s)
Affiliation
Address
email

Abstract

Denoising diffusion probabilistic models (DDPMs) [14] have shown impressive results on image and waveform generation in continuous state spaces. Here, we introduce Discrete Denoising Diffusion Probabilistic Models (D3PMs), diffusion-like generative models for discrete data that generalize the multinomial diffusion model of Hoogeboom et al. [15], by going beyond corruption processes with uniform transition probabilities. This includes corruption with transition matrices that mimic Gaussian kernels in continuous space, matrices based on nearest neighbors in embedding space, and matrices that introduce absorbing states. The third allows us to draw a connection between diffusion models and autoregressive and mask-based generative models. We show that the choice of transition matrix is an important design decision that leads to improved results in image and text domains. We also introduce a new loss function that combines the variational lower bound with an auxiliary cross entropy loss. For text, this model class achieves strong results on character-level text generation while scaling to large vocabularies on LM1B. On the image dataset CIFAR-10, our models approach the sample quality and exceed the log-likelihood of the continuous-space DDPM model.

1 Introduction

Generative modeling is a core problem in machine learning, useful both for benchmarking our ability to capture statistics of natural datasets and for downstream applications that require generating high-dimensional data like images, text, and speech waveforms. There has been a great deal of progress with the development of methods like GANs [12, 3], VAEs [19, 28], large autoregressive neural network models [37, 36, 38], normalizing flows [27, 9, 18, 26], and others, each with their own tradeoffs in terms of sample quality, sampling speed, log-likelihoods, and training stability.

Recently, diffusion models [30] have emerged as a compelling alternative for image [14, 33] and audio [5, 20] generation, achieving comparable sample quality to GANs and log-likelihoods comparable to autoregressive models with fewer inference steps. A diffusion model is a parameterized Markov chain trained to reverse a predefined forward process, which is a stochastic process constructed to gradually corrupt training data into pure noise. Diffusion models are trained using a stable objective closely related to both maximum likelihood and score matching [16, 39], and they admit faster sampling than autoregressive models by using parallel iterative refinement [24, 32, 34, 31].

Although diffusion models have been proposed in both discrete and continuous state spaces [30], most recent work has focused on Gaussian diffusion processes that operate in continuous state spaces (e.g. for real-valued image and waveform data). Diffusion models with discrete state spaces have been explored for text and image segmentation domains [15], but they have not yet been demonstrated as a competitive model class for large scale text or image generation.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

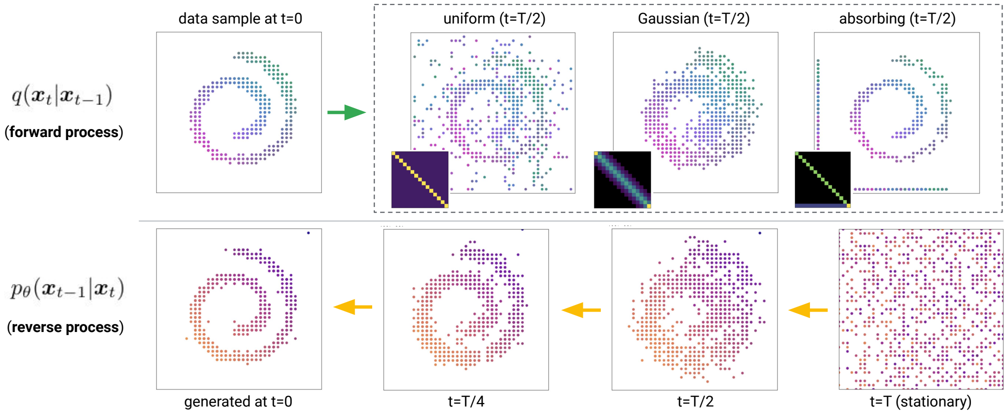


Figure 1: D3PM forward and (learned) reverse process applied to a quantized swiss roll. Each dot represents a 2D categorical variable. Top: samples from the uniform, discretized Gaussian, and absorbing state D3PM model forward processes, along with corresponding transition matrices $\boldsymbol{Q}$. Bottom: samples from a learned discretized Gaussian reverse process.

Our aim in this work is to improve and extend discrete diffusion models by using a more structured categorical corruption process to shape data generation, as illustrated in Figure 1. Our models do not require relaxing or embedding discrete data (including images) into continuous spaces, and can embed structure or domain knowledge into the transition matrices used by the forward process. We achieve significantly improved results by taking advantage of this flexibility. We develop structured corruption processes appropriate for text data, using similarity between tokens to enable gradual corruption and denoising. Expanding further, we also explore corruption processes that insert [MASK] tokens, which let us draw parallels to autoregressive and mask-based generative models. Finally, we study discrete diffusion models for quantized images, taking inspiration from the locality exploited by continuous diffusion models. This leads to a particular choice of discrete corruption process that diffuses preferentially to more similar states and leads to much better results in the image domain.

Overall, we make a number of technical and conceptual contributions. Beyond designing several new structured diffusion models, we introduce a new auxiliary loss which stabilizes training of D3PMs and a family of noise schedules based on mutual information that lead to improved performance. We strongly outperform various non-autoregressive baselines for text generation on character-level text generation, and successfully scale discrete diffusion models to large vocabularies and long sequence lengths. We also achieve strong results on the image dataset CIFAR-10, approaching or exceeding the Gaussian diffusion model from Ho et al. [14] on log-likelihoods and sample quality.

2 Background: diffusion models

Diffusion models [30] are latent variable generative models characterized by a forward and a reverse Markov process. The forward process $q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) = \prod_{t=1}^{T} q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$ corrupts the data $\boldsymbol{x}_0 \sim q(\boldsymbol{x}_0)$ into a sequence of increasingly noisy latent variables $\boldsymbol{x}_{1:T} = \boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_T$. The learned reverse Markov process $p_\theta(\boldsymbol{x}_{0:T}) = p(\boldsymbol{x}_T)\prod_{t=1}^{T} p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ gradually denoises the latent variables towards the data distribution. For example, for continuous data, the forward process typically adds Gaussian noise, which the reverse process learns to remove.

In order to optimize the generative model $p_\theta(\boldsymbol{x}_0)$ to fit the data distribution $q(\boldsymbol{x}_0)$, we typically optimize a variational upper bound on the negative log-likelihood:

$$L_{\mathrm{vb}} = \mathbb{E}_{q(\boldsymbol{x}_0)}\Bigg[\underbrace{D_{\mathrm{KL}}[q(\boldsymbol{x}_T|\boldsymbol{x}_0)||p(\boldsymbol{x}_T)]}_{L_T} + \sum_{t=2}^{T}\underbrace{\mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}\big[D_{\mathrm{KL}}[q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0)||p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)]\big]}_{L_{t-1}} \\ \underbrace{-\mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)}[\log p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1)]}_{L_0}\Bigg]. \tag{1}$$

When the number of time steps T goes to infinity, both the forward process and the reverse process share the same functional form [10], allowing the use of a learned reverse process from the same class of distributions as that of the forward process. Furthermore, for several choices of the forward process the distribution $q(\boldsymbol{x}_t|\boldsymbol{x}_0)$ converges to a stationary distribution $\pi(\boldsymbol{x})$ in the limit $t \to \infty$ independent of the value of $\boldsymbol{x}_0$. When the number of time steps T is large enough and we choose $\pi(\boldsymbol{x})$ as the prior $p(\boldsymbol{x}_T)$, we can guarantee that the L_T term in (1) will approach zero regardless of the data distribution $q(\boldsymbol{x}_0)$. (Alternatively, one can use a learned prior $p_\theta(\boldsymbol{x}_T)$.)

While $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$ can in theory be arbitrary, efficient training of p_θ is possible when $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$:

1. Permits efficient sampling of $\boldsymbol{x}_t$ from $q(\boldsymbol{x}_t|\boldsymbol{x}_0)$ for an arbitrary time t, allowing us to randomly sample timesteps and optimize each L_{t-1} term individually with stochastic gradient descent,
2. Has a tractable expression for the forward process posterior $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$, which allows us to compute the KL divergences present in the L_{t-1} term of (1).

The majority of recent work in continuous spaces [14, 31, 5, 24] defines the forward and reverse distributions as $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}\left(\boldsymbol{x}_t|\sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t\boldsymbol{I}\right)$ and $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \mathcal{N}\left(\boldsymbol{x}_{t-1}|\boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t), \boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t, t)\right)$, respectively. The aforementioned properties hold in the case of these Gaussian diffusion models: the forward process $q(\boldsymbol{x}_t|\boldsymbol{x}_0)$ converges to a stationary distribution, motivating the choice $p(\boldsymbol{x}_T) = \mathcal{N}\left(\boldsymbol{x}_T|\boldsymbol{0}, \boldsymbol{I}\right)$, and both $q(\boldsymbol{x}_t|\boldsymbol{x}_0)$ and $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ are tractable Gaussian distributions for which the KL divergence can be computed analytically.

3 Diffusion models for discrete state spaces

Diffusion models with discrete state spaces were first introduced by Sohl-Dickstein et al. [30], who considered a diffusion process over binary random variables. Hoogeboom et al. [15] extended the model class to categorical random variables with transition matrices characterized by uniform transition probabilities. In their supplementary material, Song et al. [31] also derived this extension, although no experiments were performed with this model class. Here, we briefly describe a more general framework for diffusion with categorical random variables which includes these models as special cases.

For scalar discrete random variables with K categories $x_t, x_{t-1} \in 1, ..., K$ the forward transition probabilities can be represented by matrices: $[\boldsymbol{Q}_t]_{ij} = q(x_t = j|x_{t-1} = i)$. Denoting the one-hot version of x with the row vector $\boldsymbol{x}$, we can write

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathrm{Cat}(\boldsymbol{x}_t; \boldsymbol{p} = \boldsymbol{x}_{t-1}\boldsymbol{Q}_t), \tag{2}$$

where $\mathrm{Cat}(\boldsymbol{x}; \boldsymbol{p})$ is a categorical distribution over the one-hot row vector $\boldsymbol{x}$ with probabilities given by the row vector $\boldsymbol{p}$, and $\boldsymbol{x}_{t-1}\boldsymbol{Q}_t$ is to be understood as a row vector-matrix product. We assume that $\boldsymbol{Q}_t$ is applied to each pixel of an image or each token in a sequence independently, and that q factorizes over these higher dimensions as well; we thus write $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$ in terms of a single element. Starting from $\boldsymbol{x}_0$, we obtain the following t-step marginal and posterior at time $t-1$:

$$\begin{aligned} q(\boldsymbol{x}_t|\boldsymbol{x}_0) &= \mathrm{Cat}\left(\boldsymbol{x}_t; \boldsymbol{p} = \boldsymbol{x}_0\overline{\boldsymbol{Q}}_t\right), \quad \text{with} \quad \overline{\boldsymbol{Q}}_t = \boldsymbol{Q}_1\boldsymbol{Q}_2\ldots\boldsymbol{Q}_t \\ q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) &= \frac{q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{x}_0)q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0)}{q(\boldsymbol{x}_t|\boldsymbol{x}_0)} = \mathrm{Cat}\left(\boldsymbol{x}_{t-1}; \boldsymbol{p} = \frac{\boldsymbol{x}_t\boldsymbol{Q}_t^\top \odot \boldsymbol{x}_0\overline{\boldsymbol{Q}}_{t-1}}{\boldsymbol{x}_0\overline{\boldsymbol{Q}}_t\boldsymbol{x}_t^\top}\right). \end{aligned} \tag{3}$$

Note that due to the Markov property of the forward process $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{x}_0) = q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$. Assuming that the reverse process $p_\theta(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$ is also factorized as conditionally independent over the image or sequence elements, the KL divergence between q and p_θ can be computed by simply summing over all possible values of each random variable; we thus satisfy criteria 1 and 2 discussed in Section 2. Depending on $\boldsymbol{Q}_t$, the cumulative products $\overline{\boldsymbol{Q}}_t$ can often be computed in closed form, or simply precomputed for all t. However, for large K and large T this may be prohibitive. In Appendix A.4 we discuss how to ensure $\overline{\boldsymbol{Q}}_t$ can still be computed efficiently in this case, allowing the framework to scale to a larger number of categories.

In the next section we discuss the choice of the Markov transition matrices $\boldsymbol{Q}_t$ and corresponding stationary distributions. From here on, we refer to the general class of diffusion models with discrete state spaces as Discrete Denoising Diffusion Probabilistic Models (D3PMs).

3.1 Choice of Markov transition matrices for the forward process

An advantage of the D3PM framework described above is the ability to control the data corruption and denoising process by choosing $\boldsymbol{Q}_t$, in notable contrast to continuous diffusion, for which only additive Gaussian noise has received significant attention. Besides the constraint that the rows of $\boldsymbol{Q}_t$ must sum to one to conserve probability mass, the only other constraint in choosing $\boldsymbol{Q}_t$ is that the rows of $\overline{\boldsymbol{Q}}_t = \boldsymbol{Q}_1 \boldsymbol{Q}_2 \dots \boldsymbol{Q}_t$ must converge to a known stationary distribution[1] when t becomes large, which can be guaranteed while imposing minimal restrictions on $\boldsymbol{Q}_t$ (see Appendix A.1).

We argue that for most real-world discrete data, including images and text, it makes sense to add domain-dependent structure to the transition matrices $\boldsymbol{Q}_t$ as a way of controlling the forward corruption process and the learnable reverse denoising process. Below we briefly discuss the uniform transition matrices that have been studied in prior work [15], along with a set of structured transition matrices we have explored for our image and text dataset experiments; see Appendix A.2 for more details on each matrix type. We also note that this set is not exhaustive, and many other transition matrices could also be used within the D3PM framework.

Uniform (Appendix A.2.1). Sohl-Dickstein et al. [30] considered a simple 2×2 transition matrix for binary random variables. Hoogeboom et al. [15] later extended this to categorical variables, proposing a transition matrix $\boldsymbol{Q}_t = (1 - \beta_t)\boldsymbol{I} + \beta_t / K \; \mathbb{1}\mathbb{1}^T$ with $\beta_t \in [0, 1]$. Since this transition matrix is doubly stochastic with strictly positive entries, the stationary distribution is uniform. Because the transition probability to any other state is uniform, in this paper we equivalently refer to this discrete diffusion instance as D3PM-uniform.

Absorbing state (Appendix A.2.2). Motivated by the success of BERT [8] and recent work on Conditional Masked Language Models (CMLMs) in text, we consider a transition matrix with an absorbing state (called [MASK]), such that each token either stays the same or transitions to [MASK] with some probability β_t. This does not impose particular relationships between categories, similar to uniform diffusion, but still allows corrupted tokens to be distinguished from original ones. Moreover, the stationary distribution is not uniform but has all the mass on the [MASK] token. For images, we reuse the grey pixel as the [MASK] absorbing token.

Discretized Gaussian (Appendix A.2.3). Instead of transitioning uniformly to any other state, for ordinal data we propose imitating a continuous space diffusion model by using a discretized, truncated Gaussian distribution. We choose a normalization such that the transition matrix is doubly stochastic, leading to a uniform stationary distribution. This transition matrix will transition between more similar states with higher probability, and is well suited for quantized ordinal data such as images.

Token embedding distance (Appendix A.2.4). Textual data does not have ordinal structure, but there may still be interesting semantic relationships. For instance, in a character level vocabulary vowels may be more similar to each other than they are to consonants. As a demonstration of the generality of the D3PM framework, we explore using similarity in an embedding space to guide the forward process, and construct a doubly-stochastic transition matrix that transitions more frequently between tokens that have similar embeddings while maintaining a uniform stationary distribution.

For uniform and absorbing-state diffusion, the cumulative products $\overline{\boldsymbol{Q}}_t$ can be computed in closed form (see Appendix A.4.1); the remainder can be precomputed.

3.2 Noise schedules

We consider several different options for the noise schedule of the forward process. For discretized Gaussian diffusion, we explore linearly increasing the variance of the Gaussian before discretizing it. (Note that a linear schedule for $\boldsymbol{Q}_t$ leads to a nonlinear amount of cumulative noise in $\overline{\boldsymbol{Q}}_t$.) For uniform diffusion we use the cosine schedule which sets the cumulative probability of a transition to a cosine function, as introduced by Nichol and Dhariwal [24] and adapted by Hoogeboom et al. [15]. For a general set of transition matrices $\boldsymbol{Q}_t$ (such as the one based on token embeddings), previously proposed schedules may not be directly applicable. We consider linearly interpolating the *mutual information* between $\boldsymbol{x}_t$ and $\boldsymbol{x}_0$ to zero, i.e. $I(\boldsymbol{x}_t; \boldsymbol{x}_0) \approx (1 - \frac{t}{T})\, H(\boldsymbol{x}_0)$. Interestingly, for the specific case of absorbing-state D3PMs, this schedule reduces to exactly the $(T - t + 1)^{-1}$ schedule

[1] If a stationary distribution is not known, we can instead append a rank-one matrix $\boldsymbol{Q}_{T+1}$ that ignores $\boldsymbol{x}_T$ and produces a deterministic $\boldsymbol{x}_{T+1}$; we note that this is equivalent to introducing a learned prior $p_\theta(\boldsymbol{x}_T)$.

proposed by Sohl-Dickstein et al. [30] for a Bernoulli diffusion process. See Appendix A.7 for more details.

3.3 Parameterization of the reverse process

While it is possible to directly predict the logits of $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ using a neural network $\mathrm{nn}_\theta(\boldsymbol{x}_t)$, we follow Ho et al. [14] and Hoogeboom et al. [15] and focus on using a neural network $\mathrm{nn}_\theta(\boldsymbol{x}_t)$ to predict the logits of a distribution $\widetilde{p}_\theta(\widetilde{\boldsymbol{x}}_0|\boldsymbol{x}_t)$, which we combine with $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ and a summation over one-hot representations of $\boldsymbol{x}_0$ to obtain the following parameterization

$$p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \sum_{\widetilde{\boldsymbol{x}}_0} q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \widetilde{\boldsymbol{x}}_0)\widetilde{p}_\theta(\widetilde{\boldsymbol{x}}_0|\boldsymbol{x}_t). \tag{4}$$

We note that under this $\boldsymbol{x}_0$-parameterization the KL divergence $D_{\mathrm{KL}}[q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)||p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)]$ will be zero if $\widetilde{p}_\theta(\widetilde{\boldsymbol{x}}_0|\boldsymbol{x}_t)$ places all of its probability mass on the original value $\boldsymbol{x}_0$. The decomposition of $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ in (3) also provides us with a motivation for this parameterization. According to (3), in a given state $\boldsymbol{x}_t$, the optimal reverse process only takes into account transitions to states for which $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$ is non-zero. Therefore, the sparsity pattern of $\boldsymbol{Q}_t$ determines the sparsity pattern of the ideal reverse transition probabilities in $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$. The parameterization in (4) automatically ensures that the learned reverse probability distribution $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ has the correct sparsity pattern dictated by the choice of the Markov transition matrix $\boldsymbol{Q}_t$. This parameterization also lets us perform inference with k steps at a time, by predicting $p_\theta(\boldsymbol{x}_{t-k}|\boldsymbol{x}_t) = \sum q(\boldsymbol{x}_{t-k}|\boldsymbol{x}_t, \widetilde{\boldsymbol{x}}_0)\widetilde{p}_\theta(\widetilde{\boldsymbol{x}}_0|\boldsymbol{x}_t)$.

Finally, when modeling ordinal discrete data, instead of predicting the logits of $\widetilde{p}_\theta(\widetilde{\boldsymbol{x}}_0|\boldsymbol{x}_t)$ directly with the output of a neural net, another option is to model the probabilities with a truncated discretized logistic distribution (see Appendix A.8). This provides an extra ordinal inductive bias to the reverse model and boosts FID and log-likelihood scores for images.

3.4 Loss function

While the original diffusion models introduced by Sohl-Dickstein et al. [30] were optimized with the negative variational lower bound L_{vb} of (1), more recent diffusion models are optimized with different objectives. For instance, Ho et al. [14] derive a simplified loss function (L_{simple}) that reweights the negative variational bound, and Nichol and Dhariwal [24] explore a hybrid loss $L_{\mathrm{hybrid}} = L_{\mathrm{simple}} + \lambda L_{\mathrm{vb}}$ (using one term to learn the predicted mean and the other to learn predicted variance). Inspired by this recent work, we introduce an auxiliary denoising objective for the $\boldsymbol{x}_0$-parameterization of the reverse process, which encourages good predictions of the data $\boldsymbol{x}_0$ at each time step. We combine this with the negative variational lower bound, yielding the following alternative loss function:

$$L_\lambda = L_{\mathrm{vb}} + \lambda\, \mathbb{E}_{q(\boldsymbol{x}_0)}\mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}[-\log \widetilde{p}_\theta(\boldsymbol{x}_0|\boldsymbol{x}_t)]. \tag{5}$$

Note that the auxiliary loss coincides with the cross entropy term L_0 in (1) at $t = 1$. Furthermore, due to the $\boldsymbol{x}_0$-parameterization of $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$, both the auxiliary loss term and $D_{\mathrm{KL}}[q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)||p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)]$ in L_{vb} are minimized exactly when $\widetilde{p}_\theta(\widetilde{\boldsymbol{x}}_0|\boldsymbol{x}_t)$ has all its mass on the datapoint $\boldsymbol{x}_0$. We find that training with this loss leads to improved quality of image samples.

4 Connection to existing probabilistic models for text

In this section we expand on interesting connections between the D3PM framework and several existing probabilistic and language modeling approaches.

BERT is a one-step diffusion model: One possible D3PM transition matrix is a combination of a uniform transition matrix and an absorbing state at the [MASK] token (i.e. $\boldsymbol{Q} = \alpha \mathbb{1} e_m^T + \beta \mathbb{1}\mathbb{1}^T/K + (1-\alpha-\beta)I$, where e_m is a one-hot vector on the [MASK] token). For a one-step diffusion process in which $q(\boldsymbol{x}_1|\boldsymbol{x}_0)$ replaces 10% of tokens with [MASK] and 5% uniformly at random, this leads precisely to the BERT denoising objective, i.e. $L_{vb} - L_T = -\mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)}[\log p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1)] = L_{BERT}$, since L_T is a constant independent of θ (assuming a fixed prior).

Autoregressive models are (discrete) diffusion models: Consider a diffusion process that deterministically masks tokens one-by-one in a sequence of length $N = T$: $q([\boldsymbol{x}_t]_i \mid \boldsymbol{x}_0) = [\boldsymbol{x}_0]_i$ if $i <$

$N - t$ else [MASK] . This is a deterministic forward process, so $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ is a delta distribution on the $\boldsymbol{x}_t$ sequence with one fewer mask: $q([\boldsymbol{x}_{t-1}]_i\,|\boldsymbol{x}_t, \boldsymbol{x}_0) = \delta_{[\boldsymbol{x}_t]_i}$ if $i \neq T - t$ else $\delta_{[\boldsymbol{x}_0]_i}$. While this process is not applied independently to each token, it can be recast as an independently-applied diffusion process on the product space $[0...N] \times \mathcal{V}$, where each token is tagged with its position in the sequence, $\mathcal{V}$ is the vocabulary, and $\boldsymbol{Q}$ is an $N \times |\mathcal{V}| \times N \times |\mathcal{V}|$ sparse matrix.

Because all tokens except the ith token have deterministic posteriors, the KL divergence $D_{KL}(q([\boldsymbol{x}_{t-1}]_j|\boldsymbol{x}_t, \boldsymbol{x}_0) \;||\; p_\theta([\boldsymbol{x}_{t-1}]_j|\boldsymbol{x}_t))$ is zero for all other positions. The only token for which this is not true is the token at position i, for which $D_{KL}(q([\boldsymbol{x}_{t-1}]_i|\boldsymbol{x}_t, \boldsymbol{x}_0) \;||\; p_\theta([\boldsymbol{x}_{t-1}]_i|\boldsymbol{x}_t)) = -\log p_\theta([\boldsymbol{x}_0]_i|\boldsymbol{x}_t)$, the standard cross entropy loss for an autoregressive model.

(Generative) Masked Language-Models (MLMs) are diffusion models: Generative Masked Language Models ([11], [40]) are generative models that generate text from a sequence of [MASK] tokens. They are usually trained by sampling a sequence $\boldsymbol{x}_0$, masking k tokens according to some schedule, and learning to predict the masked tokens given context. It turns out that a D3PM absorbing ([MASK]) model trained on the usual ELBO objective with the $\boldsymbol{x}_0$-parameterization from 3.3 reduces to a reweighted version of this MLM objective (see Appendix A.3 for a detailed derivation).

5 Text generation

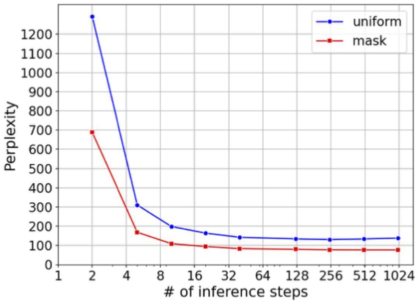
For text, we experiment with generation on two datasets: text8 [22], a character-level dataset extracted from English-language Wikipedia, and the One Billion Word dataset (LM1B) [4], a large dataset of shuffled English-language sentences. For both, we train a D3PM uniform model based on the work by Hoogeboom et al. [15] (D3PM uniform) and a model that masks tokens (D3PM absorbing). For text8, we also consider a model that transitions uniformly to nearest neighbors in a token embedding space (D3PM NN). We follow Hoogeboom et al. [15] and use $T = 1000$ timesteps, although we are also able to evaluate on fewer due to the parameterization in Section 3.3.

5.1 Character-level generation on text8

text8 is a character-level text dataset consisting of a small vocabulary of 27 tokens: the letters 'a'-'z' and the '_' whitespace token. We follow the convention of training and evaluating text8 in chunks of length 256 without any preprocessing [15]. For nearest-neighbor D3PM, our nearest neighbor graph in character-space is shown in Appendix B.2. D3PM uniform models were trained with a cosine schedule from Hoogeboom et al. [15] (ablations in Appendix B.2), while D3PM absorbing and D3PM NN models were trained with a mutual information schedule.

Table 1: Quantitative results on text8. NLL is reported on the entire test set. Sample times are for generating a single example of length 256. Results are reported on two seeds. All models are standard 12-layer transformers unless otherwise noted. †Transformer XL is a 24-layer transformer, using a 784 context window. *Results reported by [15] by running code from official repository.

Model	Model steps	NLL (bits/char) (↓)	Sample time (s) (↓)
Discrete Flow [35] (8×3 layers)	-	1.23	0.16
Argmax Coupling Flow [15]	-	1.80	0.40 ± 0.03
IAF / SCF [41]*	-	1.88	0.04 ± 0.0004
Multinomial Diffusion (D3PM uniform) [15]	1000	≤ 1.72	26.6 ± 2.2
D3PM uniform [15] (ours)	1000	$\leq 1.61 \pm 0.02$	3.6 ± 0.4
D3PM NN ($L_{\lambda=0}$) (ours)	1000	$\leq 1.59 \pm 0.03$	3.147 ± 0.0002
D3PM mask ($L_{\lambda=0.01}$) (ours)	1000	$\leq 1.45 \pm 0.02$	3.4 ± 0.3
D3PM uniform [15] (ours)	256	$\leq 1.68 \pm 0.01$	0.58 ± 0.0001
D3PM NN ($L_{\lambda=0}$) (ours)	256	$\leq 1.64 \pm 0.02$	0.81 ± 0.002
D3PM absorbing ($L_{\lambda=0.01}$) (ours)	256	$\leq 1.47 \pm 0.03$	0.59 ± 0.002
Transformer decoder (ours)	256	1.23	0.357 ± 0.0002
Transformer decoder [1]	256	1.18	-
Transformer XL [7]†	256	1.08	-
D3PM uniform [15] (ours)	20	$\leq 1.791 \pm 0.03$	0.078 ± 0.0001
D3PM NN ($L_{\lambda=0}$) (ours)	20	$\leq 1.75 \pm 0.02$	0.111 ± 0.0001
D3PM absorbing ($L_{\lambda=0.01}$) (ours)	20	$\leq 1.56 \pm 0.04$	0.078 ± 0.0003



t = 128 **[MASK] [MASK] [MASK] [MASK] [MASK] [MASK]...**
t = 25 In response **[MASK]** the demands , **[MASK] [MASK]**y Workers union said **[MASK]** backflow fund **[MASK]**s would face further investigation and a fine.
t = 0 In response to the demands , the Community Workers union said the backflow fund managers would face further investigation and a fine .

Original: Caterpillar is eager to expand in Asia , where it trails local competitors such as Komatsu Ltd
Corrupted: Caterpillar is eager to expand in**[MASK]** , **[MASK]** it **[MASK]** s local competitors such as Komatsu Ltd
Reconstructed: Caterpillar is eager to expand in China , where it faces local competitors such as Komatsu Ltd

Figure 2: Left: perplexity v.s. sampling iterations for LM1B. Right: Using a trained D3PM absorbing model for LM1B to (top) generate new sentences and (bottom) reconstruct corrupted examples.

Table 2: Quantitative results on LM1B. Perplexity reported on the test set normalized by the number of words. Results are reported on two seeds. Context window has length 128 for all diffusion models, and all have 12 layers unless otherwise noted. † Transformer XL is a 24 layer transformer.

Metric:	Perplexity			Sample time		
inference steps:	1000	128	64	1000	128	64
D3PM-uniform	137.9± 2.1	139.2 ± 1.2	145 ± 1.2	1.8 ± 0.02	0.21 ± 0.03	0.08 ± 0.01
D3PM-absorbing	76.9 ± 2.3	80.13 ± 1.2	83.62 ± 6.1	1.9 ± 0.01	0.19 ± 0.002	0.099 ± 0.01
Transformer (ours)	-	43.62549	-	-	0.256 ± 0.03	-
Transformer XL [7]†	-	21.8	-	-	-	-

Table 1 shows that for D3PM, the D3PM absorbing model performed the best, exceeding the uniform and NN diffusion models. We were able to improve upon the baseline result of [15] with hyperparameter tuning, and our uniform and NN results outperformed results from Hoogeboom et al. [15] across all inference steps, down to as few as 20. We found that $L_{\lambda=0.01}$ worked best for D3PM absorbing, while L_{vb} was better for D3PM uniform. Our model outperforms all non-autoregressive baselines except one, the Discrete Flow model [35] (for which unfortunately no open-source implementations exist), and is also faster than all but one method, the IAF/SCF model [41]. It is also nearly 20x faster than an autoregressive transformer of the same size. For further results, see Appendix B.2. D3PM with the mask absorbing token was by far the best performing model, which lends credibility to the use of masks in denoising auto-encoders. Nearest-neighbor diffusion only narrowly improves upon a D3PM-uniform model: this was a surprising negative result for us, suggesting that not all notions of structure are meaningful.

5.2 Text generation on LM1B

Text generation for large-scale text datasets and large vocabularies with discrete diffusion models has not been previously demonstrated. We include results from LM1B as a proof of concept, showing that these models can indeed scale (as discussed in Appendix A.4), and that the D3PM absorbing model continues to excel. We do not report D3PM-NN results on LM1B due to slow training and a lack of signal from the text8 results. All models were trained and evaluated on packed sequences of length 128, using a sentencepiece[2] vocabulary of size 8192.

Table 2 contains results from experiments on LM1B. We also include a plot of inference time as a function of iterations in Appendix B.3. Overall, mask diffusion (D3PM absorbing) does relatively well, approaching the performance of a comparable autoregressive model of the same size, and scaling to far fewer steps, while uniform diffusion performs significantly worse. We found the the $L_{\lambda=0.01}$ loss worked best for the mask absorbing model, but reduced performance for the other models. We note the surprising scaling in perplexity in Figure 2, achieving remarkably strong results with as few as 10 inference steps. We also show samples from our model, as well as completions from corrupted samples.

[2] https://github.com/google/sentencepiece

Table 3: Inception scores (IS), Frechet Inception Distance (FID) and negative log-likehood (NLL) on the image dataset CIFAR-10. The NLL is reported on the test set in bits per dimension. Where available, results are reported over models trained with two seeds.

Model	IS (↑)	FID (↓)	NLL (↓)
Sparse Transformer [6]			2.80
NCSN [32]	8.87 ± 0.12	25.32	
NCSNv2 [33]	8.40 ± 0.07	10.87	
StyleGAN2 + ADA [17]	9.74 ± 0.05	3.26	
Diffusion (original), L_{vb} [30]			≤ 5.40
DDPM L_{vb} [14]	7.67 ± 0.13	13.51	≤ 3.70
DDPM L_{simple} [14]	9.46 ± 0.11	3.17	≤ 3.75
Improved DDPM L_{vb} [24]		11.47	≤ 2.94
Improved DDPM L_{simple} [24]		2.90	≤ 3.37
DDPM++ cont [34]		2.92	2.99
NCSN++ cont. [34]	9.89	2.20	
D3PM uniform L_{vb}	6.09 ± 0.09	48.065 ± 0.83	$\leq 5.11 \pm 0.01$
D3PM absorbing L_{vb}	6.20 ± 0.08	41.75 ± 1.32	$\leq 4.88 \pm 0.02$
D3PM absorbing $L_{\lambda=0.001}$	6.70 ± 0.11	30.99	≤ 4.45
D3PM Gauss L_{vb}	7.92 ± 0.10	15.72	≤ 3.97
D3PM Gauss $L_{\lambda=0.001}$	8.54 ± 0.16	8.23 ± 0.18	$\leq 3.985 \pm 0.007$
D3PM Gauss + logistic $L_{\lambda=0.001}$	8.65 ± 0.19	6.90 ± 0.02	$\leq 3.442 \pm 0.003$

6 Image generation

We evaluate the performance of several D3PM models on the task of unconditional image generation with the dataset CIFAR-10 [21]. We follow Ho et al. [14] and use $T = 1000$ timesteps for all models and verify that for all models the forward process converges to the stationary distribution within T steps, yielding a value of at most $L_T \approx 10^{-5}$ bits per dimension. We train three versions of D3PM with different transition matrices: doubly stochastic matrices with uniform transition probabilities (D3PM uniform) [15], transition matrices with an absorbing state located at R, G and B values of 128 (D3PM absorbing) and doubly stochastic discretized Gaussian transition matrices (D3PM Gauss). For the D3PM uniform model we experimented with a linear β_t schedule as well as the cosine schedule as proposed in [15], with the cosine schedule producing the best results. For D3PM absorbing we use the schedule $\beta_t = (T - t + 1)^{-1}$ as also proposed in [30], which corresponds to increasing the probability of being in the absorbing state linearly over time. For D3PM Gauss we use the same linear schedule as in [14]. See Appendix B.1 for more details on the experimental setup.

Table 3 shows that for D3PM models trained with the L_{vb} objective, D3PM Gauss performs better than D3PM absorbing and uniform on all metrics: Inception score (IS), Frechet Inception Distance (FID) and negative log-likelihood (NLL). The IS score of the uniform and absorbing D3PM models

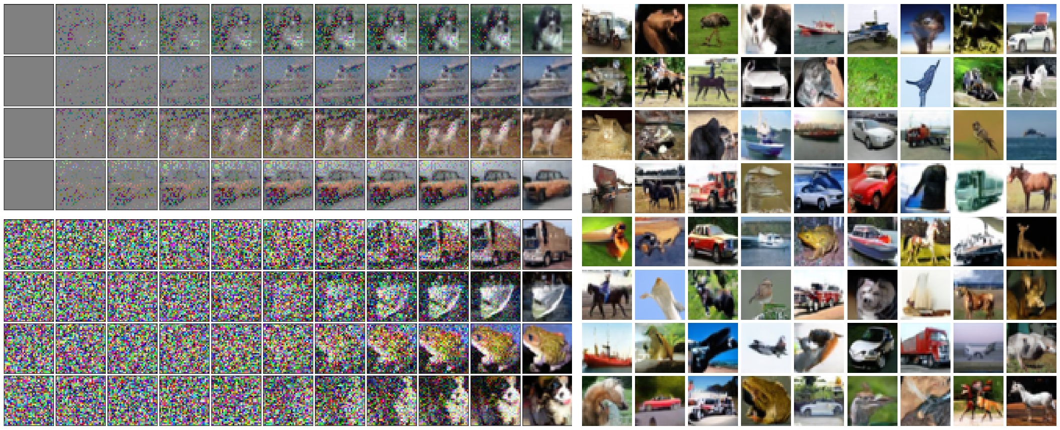


Figure 3: Left: progressive sampling at $t = 1000, 900, 800, ..., 0$ for D3PM absorbing (top) and D3PM Gauss + logistic (bottom), trained with L_λ loss on CIFAR-10. These samples were cherry picked. Right: (non cherry picked) samples from the D3PM Gauss + logistic model.

are comparable, while the FID score and NLL of the D3PM absorbing model are slightly better. We trained both D3PM absorbing and D3PM Gauss with the alternative loss function L_λ of (5), and we found $\lambda = 0.001$ to work best. We have also experimented with larger values of λ and a model trained only with the auxiliary denoising term in (5). Although this led to a more rapid increase in performance early on in training, the NLL leveled off at higher values for larger λ and the FID even started increasing again. The results show that the models trained with L_λ perform significantly better than their counterparts trained with L_{vb}. One explanation for this boost in performance is that the cross entropy term leads to gradient noise that varies less with the time step t, which is in contrast to the large change in magnitude of the L_{t-1} terms in L_{vb} for smaller t, as demonstrated by Nichol and Dhariwal [24]. Finally, we achieve our best results by combining D3PM Gauss trained on L_λ with a truncated logistic parameterization of the reverse process distribution $p_\theta(\widetilde{\boldsymbol{x}}_0|\boldsymbol{x}_t)$ (D3PM Gauss + logistic). Figure 3 shows samples from our best model (D3PM Gauss + logistic), as well as the D3PM absorbing model.

7 Related Work

Diffusion generative models were first proposed by Sohl-Dickstein et al. [30] and have gained renewed attention recently due to strong results on image and waveform generation [14, 5]. Recent works have proposed improvements for diffusion model training, including importance sampling of the ELBO, better noise schedules [24] and implicit diffusion models [31]. Several works have also drawn connections to score matching [39, 16, 32], leading to improved sampling algorithms in the continuous-time limit [34].

While most works have considered continuous diffusion models, discrete diffusion-like models were described in [30] and applied to text generation and image segmentation data in [15]. Some works [25, 23] have dealt with discrete data by embedding it in continuous space and leveraging Gaussian diffusion, but have not applied this to text. Seff et al. [29] also considered generation of discrete structured objects using a diffusion-like Markov corruption process.

For text, denoising autoencoders have a long history both in representation learning [2, 8] and more recently as generative models [40]. These closely resemble our MASK diffusion variants for a particular schedule and transition matrix (see Section 4), although our framing allows us to compute log-likelihoods and experiment with alternative transition matrices. Other works have considered non-autoregressive translation via insertion and deletion [13] or masking [11].

8 Discussion

We have presented D3PMs, a class of models that improves diffusion models for discrete data by defining new kinds of discrete corruption processes. We achieve strong empirical results relative to previous work on discrete diffusion models, even surpassing performance of continuous diffusion models in terms of log-likelihoods for image generation. While these results are promising, one limitation is that—like much other work on non-autoregressive generative models—our models are still inferior to strong autoregressive models like Transformer XL for text generation, and continuous diffusion models still yield stronger results on image quality. We expect that D3PMs can benefit further from the rapid development of continuous diffusion models [34, 24]. For example, further research in alternative losses for D3PM's can take inspiration from the reweighted L_{simple} objective used in [14], or the resampled variational bound in Nichol and Dhariwal [24]. Furthermore, D3PM's might benefit from increasing the number of timesteps and a more optimized noise schedule, as discussed in Nichol and Dhariwal [24]. Another limitation comes from the choice of evaluation metrics that we use (and that are standard for evaluation of generative models). Inception score and Frechet Inception Distance are based on neural networks that have been trained on a particular distribution of data, which is not representative for all use-cases, and focusing on average quality metrics may not accurately reflect performance across the wide diversity of settings where these generative models may be applied. This creates a risk of negative social impacts where advances disproportionately favor a subset of the population. Going forward, we are excited about the space of possibilities that arise within the D3PM framework. We have found successes in leveraging the flexibility that comes from defining discrete corruption processes for discrete data, but we believe that there are many more possibilities that make use of richer forms of structure to define even more powerful discrete diffusion models.

References

[1] Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. Character-Level language modeling with deeper Self-Attention. August 2018.

[2] Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. Generalized denoising Auto-Encoders as generative models. May 2013.

[3] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019.

[4] Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. One billion word benchmark for measuring progress in statistical language modeling. December 2013.

[5] Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. WaveGrad: Estimating gradients for waveform generation. September 2020.

[6] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

[7] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a Fixed-Length context. January 2019.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. October 2018.

[9] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using Real NVP. *arXiv preprint arXiv:1605.08803*, 2016.

[10] W Feller. On the theory of stochastic processes, with particular reference to applications. In *Proceedings of the [First] Berkeley Symposium on Mathematical Statistics and Probability*. The Regents of the University of California, 1949.

[11] Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. Mask-Predict: Parallel decoding of conditional masked language models. April 2019.

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[13] Jiatao Gu, Changhan Wang, and Jake Zhao. Levenshtein transformer. May 2019.

[14] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, pages 6840–6851, 2020.

[15] Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Towards non-autoregressive language models. *arXiv preprint arXiv:2102.05379*, 2021.

[16] Aapo Hyvärinen, Juha Karhunen, and Erkki Oja. *Independent component analysis*, volume 46. John Wiley & Sons, 2004.

[17] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *arXiv preprint arXiv:2006.06676v1*, 2020.

[18] Diederik P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, pages 10215–10224, 2018.

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

[20] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*, 2020.

[21] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[22] Matt Mahoney. Text8 dataset. `http://mattmahoney.net/dc/textdata`, 2011. Accessed: 2021-5-24.

[23] Gautam Mittal, Jesse Engel, Curtis Hawthorne, and Ian Simon. Symbolic music generation with diffusion models. March 2021.

[24] Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. 2021.

[25] Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, and Stefano Ermon. Permutation invariant graph generation via score-based generative modeling. March 2020.

[26] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *arXiv preprint arXiv:1912.02762*, 2019.

[27] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning*, pages 1530–1538, 2015.

[28] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, pages 1278–1286, 2014.

[29] Ari Seff, Wenda Zhou, Farhan Damani, Abigail Doyle, and Ryan P Adams. Discrete object generation with reversible inductive construction. July 2019.

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

[31] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

[32] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, pages 11895–11907, 2019.

[33] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *arXiv preprint arXiv:2006.09011*, 2020.

[34] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. November 2020.

[35] Dustin Tran, Keyon Vafa, Kumar Agrawal, Laurent Dinh, and Ben Poole. Discrete flows: Invertible generative models of discrete data. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[36] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. WaveNet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[37] Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. *International Conference on Machine Learning*, 2016.

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

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

[40] Alex Wang and Kyunghyun Cho. BERT has a mouth, and it must speak: BERT as a markov random field language model. February 2019.

[41] Zachary M Ziegler and Alexander M Rush. Latent normalizing flows for discrete sequences. January 2019.

Checklist

1. For all authors...
   - (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   - (b) Did you describe the limitations of your work? [Yes] See Section 8
   - (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 8
   - (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   - (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   - (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   - (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] The code cannot be made available at this time, but we will work to get it open sourced for a camera ready version.
   - (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] The training details can be found in Appendix B.
   - (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We have included averages and standard deviations for as many models as possible for two seeds.
   - (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix B.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   - (a) If your work uses existing assets, did you cite the creators? [Yes] We have included citations to all datasets.
   - (b) Did you mention the license of the assets? [N/A] These are standard datasets.
   - (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   - (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   - (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] While it is valid to question if such information appears in large datasets like CIFAR-10 and LM1B, these are standard benchmarks, and it is beyond the scope of this work to do a thorough analysis.
5. If you used crowdsourcing or conducted research with human subjects...
   - (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   - (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   - (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]