Diffusing Gaussian Mixtures for Generating Categorical Data

Florence Regol, Mark Coates

Dept. Electrical and Computer Engineering, McGill University
Montréal, QC, Canada
florence.robert-regol@mail.mcgill.ca, mark.coates@mcgill.ca

Abstract

Learning a categorical distribution comes with its own set of challenges. A successful approach taken by state-of-the-art works is to cast the problem in a continuous domain to take advantage of the impressive performance of the generative models for continuous data. Amongst them are the recently emerging diffusion probabilistic models, which have the observed advantage of generating high-quality samples. Recent advances for categorical generative models have focused on log likelihood improvements. In this work, we propose a generative model for categorical data based on diffusion models with a focus on high-quality sample generation, and propose sampled-based evaluation methods. The efficacy of our method stems from performing diffusion in the continuous domain while having its parameterization informed by the structure of the categorical nature of the target distribution. Our method of evaluation highlights the capabilities and limitations of different generative models for generating categorical data, and includes experiments on synthetic and real-world protein datasets.

Introduction

There are numerous applications for generative models of categorical random sequences; text generation, speech and music synthesis, drug design and protein synthesis are all important tasks that require modeling of high-dimensional nominal data. Learning the structure and substructure underlying those complex high-dimensional distributions can be useful for downstream tasks. For example, in drug synthesis, studies have confirmed the important role that mutational covariation plays in determining protein function, and this has found practical applications in drug design and drug resistance prediction (McGee et al. 2021; Tubiana, Cocco, and Monasson 2019; Socolich et al. 2005). As a result, recent works have employed generative models to learn from existing proteins and generate new ones (Trinquier et al. 2021; McGee et al. 2021; Jain et al. 2022). For this type of problem, the ability to generate quality samples is essential.

While the research on generative models for continuous data has been flourishing (see (Bond-Taylor et al. 2022) for a review), the literature on modeling nominal categorical data is not as developed (Hoogeboom et al. 2021a).

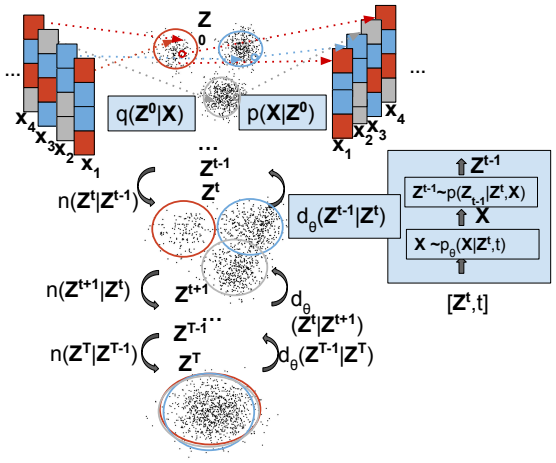

Figure 1: Overview of the architecture. On the left, a visualisation of the diffusion process for the first element of a sequence $\mathbf{x}_{(1)}$ is depicted. The sequence is mapped to the continuous space through the fixed Gaussian Mixture (GM) encoder $q(\mathbf{Z}^0|\mathbf{X} = \mathbf{x})$, then is diffused through the iterative application of noise distributions $n(\mathbf{Z}^t|\mathbf{Z}^{t-1})$ until the signal is destroyed at $\mathbf{Z}^T$. The right depicts the generative process. Starting from $\mathbf{Z}^T$, the denoising function models a distribution $\mathbf{x}$ conditioned on $\mathbf{z}^T, t$ which in turns models the mixture component of the fixed GM that will be used to produce $\mathbf{Z}^{t-1}$ conditioned on $\mathbf{Z}^t, t$. The final sequence is generated from the decoder $p(\mathbf{X}|\mathbf{Z}^0)$.

Autoregressive (AR) methods are well suited for modeling categorical data (Cooijmans et al. 2017). A notable class of AR models that give impressive performance for this problem are Transformers (Dai et al. 2019; Child et al. 2019; Hua et al. 2022; Jun et al. 2020). Transformers are powerful, but generally suffer from the weaknesses associated with autoregressive models; they are generally slow to train and slow to sample from (Bond-Taylor et al. 2022). They also suffer from quadratic complexity (w.r.t. sequence length), and because of their impressive flexibility in modeling capability, are harder to apply to smaller datasets (Lin et al. 2021). As a result, many works have attempted to linearize the time/memory complexity (Hua et al. 2022; Katharopou-

los et al. 2020; Kitaev, Kaiser, and Levskaya 2020), but these limitations still remain key challenges (Lin et al. 2021).

Discretization of continuous methods has been explored (Dinh, Sohl-Dickstein, and Bengio 2017; Ho et al. 2019; Theis, van den Oord, and Bethge 2016; Uria, Murray, and Larochelle 2013). However the modeling assumptions of these works are not suited to data that has no natural ordering of the categories. For the specific problem of nominal data generation, current state-of-the-art works are based on extending generative models that were initially developed for continuous data: normalizing flow (Ziegler and Rush 2019; Lippe and Gavves 2021; Hoogeboom et al. 2021b) and diffusion models (Hoogeboom et al. 2021a). Hoogeboom et al. report results indicating that the diffusion models can outperform Transformers. Diffusion probabilistic models (Sohl-Dickstein et al. 2015) are attractive for their generative capability. Compared to their competitors, such models have the characteristic of generating high quality samples and are relatively fast to train. The general trade-off is that they achieve lower likelihood and slower sampling (Bond-Taylor et al. 2022; Ho, Jain, and Abbeel 2020). As a result, substantial effort has been devoted to address these limitations (Nichol and Dhariwal 2021; Kingma et al. 2021; Xiao, Kreis, and Vahdat 2022; Salimans and Ho 2022).

In this work, we propose a generative model based on a diffusion process that can remain in the continuous space without sacrificing our knowledge that the data is nominal. To do so, we introduce a novel approach to encode nominal data in the continuous space via a sphere packing algorithm that places each category in the encoding space. We then incorporate the structural knowledge that follows from this construction into the denoising step of the diffusion using a Gaussian mixture conditioned on the current state of the diffusion. The advantages of such a design are threefold: 1) Unlike previous work (Hoogeboom et al. 2021a; Lippe and Gavves 2021; Hoogeboom et al. 2021b), this fixed encoding allows flexibility of the dimensionality of the representations without added complexity; 2) the structured denoising step requires significantly fewer diffusion steps, which greatly improves sampling time (which is identified as one of the main limitations of the diffusion model) while keeping the benefit of the diffusion model; 3) the generated samples are of higher quality.

Currently, the main method of evaluating a categorical generative model is via the log likelihood of held-out data. Although useful, this metric has some known drawbacks. (Theis, van den Oord, and Bethge 2016) use a simple example to show clearly how a good likelihood does not guarantee good sample generation. Proper evaluation of generative models is an ongoing research topic in many fields, including text, image, and graph generation (Garbacea et al. 2019; Celikyilmaz, Clark, and Gao 2020; Zhou et al. 2019; Borji 2019; Thompson et al. 2022; Theis, van den Oord, and Bethge 2016; Wu et al. 2017).

The general consensus has been to push towards a more comprehensive and task-oriented approach for assessing performance. Candidate metrics do not necessarily correlate with each other (Theis, van den Oord, and Bethge 2016; Zhou et al. 2019), so it can be important to measure performance in multiple ways. Indeed, failure to follow a comprehensive evaluation methodology has been linked to difficulties in assessing which models are actually better and to unexpected results (Caccia et al. 2020; Lucic et al. 2018; Rabanser, Günnemann, and Lipton 2019). A notable example is the finding by (Nagarajan, Andreassen, and Neyshabur 2021) that high likelihood on a dataset and good sample generation does not guarantee good out-of-distribution detection capability, one of the candidate uses of a good generative model.

With these observations in mind, in this work, we expand on standard evaluation metrics to include distribution distance metrics. We propose a synthetic experiment with a known ground truth distribution to aid performance evaluation, with the goal of providing a more complete account of the generative capability of the models considered. To summarize, the major contributions of this paper are:

1. We introduce a **novel procedure to represent nominal data in the continuous space** based on sphere packing.
2. This allows us to design a **novel denoising function tailored** to model nominal data in the continuous space.
3. Our presented model offers state-of-the-art **sample generation quality** and is efficient in both **sampling time** and **training time**, as demonstrated by our experiments on both synthetic datasets and on protein datasets.

Related Work

Early approaches to handle the related problem of discrete data generation were based on dequantization and thresholding. The overall idea is to add noise to the discrete point and treat it as a continuous generative modeling problem, and then use thresholding to generate samples (Ho et al. 2019; Theis, van den Oord, and Bethge 2016; Dinh, Sohl-Dickstein, and Bengio 2017). Current state-of-the-art methods avoid injecting an arbitrary ordering to the categories by either adapting the methodology to stay in the categorical domain, or modelling the data using a latent representation in the continuous space that can be later mapped to the categorical space. In (Ziegler and Rush 2019) and (Lippe and Gavves 2021), normalizing flows (NF) are used to model such a latent representation. An encoder-decoder framework is used to map from the categorical to the continuous space and vice versa. The overall model is learned through variational inference. (Hoogeboom et al. 2021b) build on the same idea as (Lippe and Gavves 2021), but rather than learning the encoder/decoder, they fix the decoder with an argmax function. This induces a constraint on the functional space of the encoder that is maintained throughout training. Both of these state-of-the-art works keep the mapping from the continuous to the categorical space simple. In (Hoogeboom et al. 2021a) this is done by using a fixed deterministic argmax function, and (Lippe and Gavves 2021) experimented with learning the encoder/decoder of varying complexity and found that a simple parameterization of the mean and variance gave the optimal result. Unlike our approach, once this mapping is done, nothing informs the NF that it is treating a latent representation of a categorical variable.

Moving away from the normalizing flow methods, (Hoogeboom et al. 2021a) also presented a diffusion-based model that operates directly on the categorical space. Instead of diffusing the signal with Gaussian distributions and learning means and variance of parameterized Gaussian as denoising process, they diffuse a one-hot encoded sequence with a multinomial categorical distribution. As is the case for the argmax, the dimension of the sequence representation scales linearly with the number of categories. Other related work that takes a similar approach to us by mapping to an alternative space to perform diffusion includes (Vahdat, Kreis, and Kautz 2021) and (Sinha et al. 2021). These works tackle the tangential problems of generating ordinal data and conditional generative modeling.

Lastly, related works that target a similar task connected to generating quality proteins include (Jain et al. 2022; Brookes, Park, and Listgarten 2019; Kumar and Levine 2020; Hoffman et al. 2022). This literature focuses on generating high score protein sequences, which are evaluated by an oracle. Even though these models are generative in nature, the end task is still somewhat supervised. The models explicitly aim to maximize a quantity, whereas for our purposes we remain in the traditional generative modeling problem formulation of learning a distribution.

Methodology

Problem Setting. Consider a categorical multivariate random variable $\mathbf{X} = [X_{(1)}, \dots X_{(S)}]$ where each element belongs to one of K categories: $X_{(j)} \in \mathcal{C}, \mathcal{C} = \{C_1, \dots, C_K\}$ with associated pmf $p(\mathbf{X})$. Given a dataset of realizations $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^N, \mathbf{x}_i \sim \mathbf{X}$, the task is to learn $p(\mathbf{X})$.

Encoding the categorical sequences and sphere packing. We lift the problem to the continuous space by introducing a latent continuous random variable $\mathbf{Z}^0$ that is mapped from and to the categorical sequence $\mathbf{X} \in \mathcal{C}^S$ with an encoder, $q(\mathbf{Z}^0|\mathbf{X})$, and decoder, $p(\mathbf{X}|\mathbf{Z}^0)$, respectively. The log likelihood and its variational lower bound are given by:

$$\log p(\mathbf{X}) = \log \int \frac{p(\mathbf{X}, \mathbf{Z}^0)}{q(\mathbf{Z}^0|\mathbf{X})} q(\mathbf{Z}^0|\mathbf{X})\, d\mathbf{Z}^0,$$

$$\log p(\mathbf{X}) \geq \mathbb{E}_{q(\mathbf{Z}^0|\mathbf{X})}\Big[\log\Big(p(\mathbf{Z}^0)\Big) + \log\Big(\frac{p(\mathbf{X}|\mathbf{Z}^0)}{q(\mathbf{Z}^0|\mathbf{X})}\Big)\Big]. \quad (1)$$

It is desirable to focus complexity into learning $p(\mathbf{Z}^0)$, so we make the mappings from $\mathbf{Z}^0$ to $\mathbf{X}$ simple and tractable. Consequently, we use a fixed, factorized encoding distribution to associate each categorical element $X_{(s)}$ of the sequence with a random vector in a d-dimensional continuous space $\mathbf{Z}^0_{(s)} \in \mathbb{R}^d$. The mapping depends on the category; each category C_k is assigned a distribution $f(\cdot; \boldsymbol{\mu}_{C_k}, \sigma)$ that is clearly distinguishable from others by its mean $\boldsymbol{\mu}_{C_k} \in \mathbb{R}^d$ and variance $\sigma^2 \in \mathbb{R}$. We use a Gaussian $f(\cdot)$ for simplicity, and similarly to (Lippe and Gavves 2021) we obtain the decoder $p(\mathbf{X}|\mathbf{Z}^0)$ through Bayes' rule, so we have:

$$q(\mathbf{Z}^0|\mathbf{X}) = \prod_{s=1}^{S} \mathcal{N}(\mathbf{Z}^0_{(s)}; \boldsymbol{\mu}_{X_{(s)}}, \mathbf{I}\sigma^2) \text{ as the encoder and}$$

$$p(\mathbf{X}|\mathbf{Z}^0) = \prod_{s=1}^{S} \frac{\mathcal{N}(\mathbf{Z}^0_{(s)}; \boldsymbol{\mu}_{X_{(s)}}, \mathbf{I}\sigma^2)}{\sum_{k=1}^{K} \mathcal{N}(\mathbf{Z}^0_{(s)}; \boldsymbol{\mu}_{C_k}, \mathbf{I}\sigma^2)} \text{ as the decoder.}$$

(The prior on $\mathbf{X}$ does not appear as we assume uniformity). The advantages are twofold: 1) it imposes a structure on the target distribution $p(\mathbf{Z}^0)$ that can be used in modeling the learnable $p_\theta(\mathbf{Z}^0)$, as we will show shortly; and 2) it simplifies the learning objective since only $p(\mathbf{Z}^0)$ is learnable.

Our aim is to make it as easy as possible for the decoder to distinguish between categories. This implies that we should strive to identify maximally separated means. This leads to a sphere packing problem — finding the emplacement of K points on the surface of a d-dimensional sphere $\mathbb{S}^d(1)$ that maximizes the minimum distance between any two points:

$$\boldsymbol{\mu}_1^*, \dots, \boldsymbol{\mu}_K^* = \underset{\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K \in \mathbb{S}^d(1)}{\arg\max} \quad \Big(\min_{i \neq j} ||\boldsymbol{\mu}_i - \boldsymbol{\mu}_j||_2^2\Big)$$

Hence we can use solutions of this problem, e.g., (Gamal et al. 1987), to 1) set the means of the encoding distributions $\{\boldsymbol{\mu}_{C_k}\}_{k=1}^K$; and 2) determine, based on the minimum distance $d_{\boldsymbol{\mu}^*} = \min_{i \neq j} ||\boldsymbol{\mu}_i^* - \boldsymbol{\mu}_j^*||_2^2$, a value for the variance σ^2 such that the Gaussian distributions $\mathcal{N}(\boldsymbol{\mu}_{C_k}, \mathbf{I}\sigma^2); k \in [K]$ have limited overlap but are not too concentrated. Denoting $d_{\boldsymbol{\mu}^*} = \min_{i \neq j} ||\boldsymbol{\mu}_i^* - \boldsymbol{\mu}_j^*||_2^2$, we have:

$$\boldsymbol{\mu}_{C_k} = \boldsymbol{\mu}_k^*; \quad k \in [K] \quad \text{and} \quad \sigma = \frac{d_{\boldsymbol{\mu}^*}}{2K\sqrt[d]{3}}. \quad (2)$$

Almost all (99.7%) of the mass of a d-dimensional m.v. Gaussian R.V. is within $\sqrt[d]{3}$ standard deviations, so we set σ to half that radius, and divide by the number of categories.

Learning the latent distributions $p_\theta(\mathbf{Z}^0)$. The complex correlation structure of the categorical distribution must be captured in $p_\theta(\mathbf{Z}^0)$. We propose to use a diffusion probabilistic model (DPM) (Sohl-Dickstein et al. 2015) with a novel denoising component, tailored to our encoding scheme and categorical data, based on Gaussian Mixtures. The DPM introduces T latent random variables $\mathbf{Z}^1, \dots, \mathbf{Z}^T$. Commencing with the targeted encoded sequence $\mathbf{Z}^0$, the variables are derived by gradually adding known Gaussian noise of increasing variance to the variable from the previous timestep: $n(\mathbf{Z}^t|\mathbf{Z}^{t-1}) = \mathcal{N}(\mathbf{Z}^t; \sqrt{1-\beta_t}\mathbf{Z}^{t-1}, \beta_t \mathbf{I}); \beta_i < \beta_{i+1} \in (0, 1)$. At the end of the chain, only noise should remain $\mathbf{Z}^T \sim \mathcal{N}(\mathbf{Z}^T; \mathbf{0}, \mathbf{1})$. The task of the DPM is to learn the denoising process $d_\theta(\mathbf{Z}^{t-1}|\mathbf{Z}^t); t \in [T]$.

This leads to construction of the generative model for $\mathbf{Z}^0$:

$$p_\theta(\mathbf{Z}^0) \overset{\text{m}}{=} d_\theta(\mathbf{Z}^0) = \int d(\mathbf{Z}^T) \prod_{t=1}^{T} d_\theta(\mathbf{Z}^{t-1}|\mathbf{Z}^t) d\mathbf{Z}^{1:T}. \quad (3)$$

See (Sohl-Dickstein et al. 2015; Ho, Jain, and Abbeel 2020) for more detailed discussion of the diffusion process.

Exploiting the structure. In most denoising approaches, the distributions $d_\theta(\mathbf{Z}^{t-1}|\mathbf{Z}^t)$ are modelled as normal distributions with learnable means and (usually fixed) variances.

In our case, we take advantage of the known structure. By our construction, the target distribution is a mixture of Gaussians; conditioned on knowledge of the target sequence, the distribution $p(\mathbf{Z}^{t-1}|\mathbf{Z}^t, \mathbf{X})$ is Gaussian, and the mean and variance can be evaluated analytically.

If we are at a point in the chain $\mathbf{z}^t$, then if we are given an element of the sequence $x_{(s)}$, $\mathbf{Z}^{t-1}_{(s)}$ is conditionally independent of other $\mathbf{Z}^{t-1}_{(s')}$, and we can derive the conditional of the next denoising step in closed-form:

$$\begin{aligned} p(\mathbf{Z}^{t-1}_{(s)}|\mathbf{Z}^t_{(s)}, x_{(s)}) &= \int p(\mathbf{Z}^{t-1}_{(s)}|\mathbf{Z}^t_{(s)}, \mathbf{Z}^0_{(s)}) p(\mathbf{Z}^0_{(s)}|x_{(s)}) d\mathbf{Z}^0_{(s)} \\ &= \mathcal{N}(\mathbf{Z}^{t-1}_{(s)}; \boldsymbol{\mu}^{\mathbf{Z}^t,t}_{x_{(s)}}, \mathbf{I}\sigma_t^2) \qquad (4) \\ \text{where } \boldsymbol{\mu}^{\mathbf{Z}^t,t}_{x_{(s)}} &= \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\boldsymbol{\mu}_{x_{(s)}} + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{Z}^t_{(s)}, \\ \sigma_t^2 &= \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t + (\frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\sigma)^2. \end{aligned}$$

(See the supplementary for the detailed derivation.) Hence if we have a predictor $p_\theta(\mathbf{X}|\mathbf{Z}^t, t)$ of the distribution of the sequence $\mathbf{X}$ based on the current state $\mathbf{z}^t$ and the diffusion step t, we can model the denoising step as:

$$d_\theta(\mathbf{Z}^{t-1}|\mathbf{Z}^t) = \sum_{\mathbf{X}\in\mathcal{C}^S} \left(\prod_{s=1}^{S} p(\mathbf{Z}^{t-1}_{(s)}|\mathbf{Z}^t_{(s)}, X_{(s)}) \right) p_\theta(\mathbf{X}|\mathbf{Z}^t, t)\,. \qquad (5)$$

If $p_\theta(\mathbf{X}|\mathbf{Z}^t, t)$ is structured to assume independence among the elements of $\mathbf{X}$, we can factorize $p_\theta(\mathbf{X}|\mathbf{Z}^t, t) = \prod_{s=1}^{S} p_\theta(X_{(s)}|\mathbf{Z}^t, t)$ and write:

$$\begin{aligned} d_\theta(\mathbf{Z}^{t-1}|\mathbf{Z}^t) &= \prod_{s=1}^{S}\sum_{k=1}^{K} p(\mathbf{Z}^{t-1}_{(s)}|\mathbf{Z}^t_{(s)}, C_k) p_\theta(X_{(s)} = C_k|\mathbf{Z}^t, t) \\ &= \prod_{s=1}^{S} d_{(s),\theta}(\mathbf{Z}^{t-1}_{(s)}|\mathbf{Z}^t). \qquad (6) \end{aligned}$$

Replacing the Gaussian denoising term used in (Sohl-Dickstein et al. 2015; Ho, Jain, and Abbeel 2020) with this more complex denoising model results in a more involved loss expression, but the denoising process can be successful with far fewer diffusion steps (10-40 versus thousands). This effect was also observed in (Xiao, Kreis, and Vahdat 2022).

Loss objective. Since the encoder and decoder are fixed, optimization of the loss function (Eqn. (1)) simplifies to:

$$\theta^* = \arg\max_{\theta\in\Theta} \mathbb{E}_{q(\mathbf{Z}^0|\mathbf{X})}\Big[\log\Big(p_\theta(\mathbf{Z}^0)\Big)\Big], \qquad (7)$$

i.e., the log likelihood of the diffusion model under the expectation of the encoder. Since the DPM is a latent variable model, its log likelihood is also optimized via a lower bound:

$$\begin{aligned} \log\Big(p_\theta(\mathbf{Z}^0)\Big) \geq &= \mathbb{E}_n\Big[-\log\frac{n\left(\mathbf{Z}^T \mid \mathbf{Z}^0\right)}{d\left(\mathbf{Z}^T\right)} \\ &- \sum_{t=2}^{T}\log\frac{n\left(\mathbf{Z}^{t-1} \mid \mathbf{Z}^t, \mathbf{Z}^0\right)}{d_\theta(\mathbf{Z}^{t-1}|\mathbf{Z}^t)} + \log d_\theta\left(\mathbf{Z}^0 \mid \mathbf{Z}^1\right)\Big]. \end{aligned}$$

Employing this bound in Eqn. (7), substituting with (6), and removing terms that are independent of θ, we can identify the final optimization task:

$$\theta^* = \arg\max_{\theta\in\Theta} \mathbb{E}_{q,n}\Big[-\sum_{t=2}^{T}\mathcal{L}_{t-1} + \mathcal{L}_0\Big] \qquad (8)$$

where $\mathcal{L}_{t-1} = KL\Big(n\left(\mathbf{Z}^{t-1} \mid \mathbf{Z}^t, \mathbf{Z}^0\right)||d_\theta(\mathbf{Z}^{t-1}|\mathbf{Z}^t)\Big)$ and $\mathcal{L}_0 = \log d_\theta\left(\mathbf{Z}^0 \mid \mathbf{Z}^1\right)$.

Architecture and training. In practice, it has been shown beneficial for this type of loss to randomly optimize one of the terms $\mathcal{L}_t$ at a time (Ho, Jain, and Abbeel 2020) (Nichol and Dhariwal 2021). The objective then becomes to either maximize the log likelihood of the final step for $t = 0$, or to minimize the KL divergence between a Gaussian mixture with learnable mixture weights for time step $t > 0$:

$$\begin{aligned} \mathcal{L}_{t-1} &= KL\Big(n\left(\mathbf{Z}^{t-1} \mid \mathbf{Z}^t, \mathbf{Z}^0\right)||d_\theta(\mathbf{Z}^{t-1}|\mathbf{Z}^t)\Big) \\ &= \sum_{s=1}^{S} KL\Big(n\left(\mathbf{Z}^{t-1}_{(s)} \mid \mathbf{Z}^t_{(s)}, \mathbf{Z}^0_{(s)}\right)||d_{(s),\theta}(\mathbf{Z}^{t-1}_{(s)}|\mathbf{Z}^t)\Big) \end{aligned}$$

Using the variational approximation of the KL divergence between Gaussian mixtures from (Hershey and Olsen 2007), we can approximate the individual step loss as follow:

$$\begin{aligned} \mathcal{L}_{t-1} &\approx -\sum_{s=1}^{S}\log\sum_{k=1}^{K} p_\theta(X_{(s)} = C_k|\mathbf{Z}^t, t) w^s_{\mathbf{Z}^{t,0}}(C_k) \\ \text{where } w^s_{\mathbf{Z}^{t,0}}(C_k) &= \exp^{-KL\left(n(\cdot|\mathbf{Z}^t_{(s)}, \mathbf{Z}^0_{(s)})||\mathcal{N}(\cdot;\boldsymbol{\mu}^{\mathbf{Z}^t,t}_{C_k}, \sigma_t^2\mathbf{I})\right)}. \end{aligned}$$

Details of the derivation are provided in the supplementary.

At this point, we can see that the optimization of this term is reached when $p_\theta(X_{(s)}|\mathbf{Z}^t, t)$ gives maximum weight to the highest term of the sum $w^s_{\mathbf{Z}^{t,0}}(C_k)$, which is the initial sequence $C_k = x_{(s)}$. As a result, we approximate this optimization by maximizing the log likelihood of $p_\theta(X_{(s)} = x_{(s)}|\mathbf{Z}^t, t)$, as both isolated optimization problems have the same solution:

$$\arg\max_{\theta\in\Theta} \mathcal{L}_{t-1} \approx \arg\max_{\theta\in\Theta} \log p_\theta(\mathbf{X} = \mathbf{x}|\mathbf{Z}^t, t) \qquad (9)$$

As a result, learning hinges on the modeling capability of $p_\theta(\mathbf{X}|\mathbf{Z}^t, t)$. We employ a transformer-based architecture. The vector $\mathbf{z}^t$ and an embedding of time t serve as inputs. We adopt a sampling approach for the training. For each sequence in the training data, we sample $\mathbf{z}^0 \sim q(\mathbf{Z}^0|\mathbf{X} = \mathbf{x})$, and then draw a time t, we sample $\mathbf{z}^t \sim n(\mathbf{Z}^t|\mathbf{z}^0)$ to evaluate the loss. It is important to emphasize that this transformer does *not* have an autoregressive structure — all elements of a sequence are generated in parallel. The correlations are induced by the denoising diffusion process.

Data Augmentation. In practice, we observe that $p_\theta(\mathbf{X}|\mathbf{Z}^t, t)$ learns to be increasingly certain of its prediction as we approach the end of the chain $\mathbf{Z}^0$. This behavior can be seen in Figure 2 where we show an example of the entropy at every time step $H(p_\theta(\mathbf{X}|\mathbf{Z}^T, T)), \ldots, H(p_\theta(\mathbf{X}|\mathbf{Z}^1, 1))$.

We can imagine that alongside the gradually noisy $\mathbf{Z}^t$, there is also a corresponding noisy categorical sequence $\tilde{\mathbf{X}}^t$

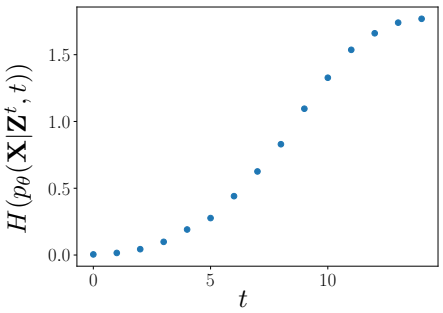

Figure 2: Average entropy of $p_\theta(\mathbf{X}|\mathbf{Z}^t, t)$ along the diffusion process during sampling. The model becomes more and more certain (lower entropy) as we approach $t = 0$.

that $p_\theta(\mathbf{X}|\mathbf{Z}^t, t)$ aims to predict. As a result, instead of training on the ground truth sequence at the beginning of the diffusion $T, T-1, \ldots$, we inject some noise by training on a "diffused" version of $\mathbf{x}$, denoted by $\tilde{\mathbf{x}}^t$, and thus modify Eqn. (9) to:

$$\arg\max_{\theta\in\Theta} \mathcal{L}_{t-1} \approx \arg\max_{\theta\in\Theta} \log p_\theta(\mathbf{X} = \tilde{\mathbf{x}}^t|\mathbf{Z}^t, t) \tag{10}$$

$$\tilde{\mathbf{x}}^t \sim p^{\mathbf{z}^t,t}(\tilde{\mathbf{X}}), \text{ where } p^{\mathbf{z}^t,t}(\tilde{\mathbf{X}})_s = \frac{(w^s_{\mathbf{Z}^{t,0}}(x_{(s)}))^\omega}{\sum_{k=1}^K (w^s_{\mathbf{Z}^{t,0}}(C_k))^\omega}$$

Algorithms detailing the training and sampling procedures are provided in the supplementary.

Experiments

In this section, we first present the evaluation metrics, the datasets and the experimental set-up. We then report the performance of our proposed GMCD model and conduct ablation studies to validate the effectiveness of its modules.

Evaluation metrics. Many of the difficulties and limitations associated with evaluating generative models stem from the fact that we do not have access to the ground truth distribution. With access to ground truth, the problem formulation changes and the previously mentioned problems associated with log likelihood (LL) and sampled-based metrics disappear. Instead of:

- Maximizing the LL of unseen samples $\rightarrow$, we aim to assign the correct probability mass to unseen samples,
- Generating "good" samples $\rightarrow$ we aim to generate samples that are distributed according to the ground truth,
- Maximizing a heuristic for sample quality (novelty, diversity, etc.) $\rightarrow$ we aim to generate samples with the same heuristic value as the expected value from ground truth.

In this work, we are interested in evaluating how close a generative model is to the true probability measure based on its samples in the discrete domain.

With known ground truth distribution. The distance between two distributions p, q on a discrete sample space Ω can be measured by the total variation and Hellinger distances:

$$d_{TV}(p,q) \triangleq \frac{1}{2}||p-q||_1 = \frac{1}{2}\sum_{x\in\Omega}|p_x - q_x|,$$

$$Hel(p,q) \triangleq \frac{1}{\sqrt{2}}||\sqrt{p}-\sqrt{q}||_2 = \frac{1}{\sqrt{2}}\sqrt{\sum_{x\in\Omega}(\sqrt{p_x}-\sqrt{q_x})^2}.$$

(with p_x used as a shorthand for $p(x)$). These are principled metrics but they can rapidly become impractical as Ω grows, especially as we must usually rely on samples to estimate p_x. Alternatively, we can consider a partitioning of the sample space: $\mathcal{P} = \{A_i; A_i \subset \Omega, A_i \cap A_j = \emptyset\}$ and estimate the probability mass of these events $p(A_i) = \sum_{x\in A_i} p_x$. It is less precise but can be more informative if Ω is large and/or if the partitioning has a particular meaning. One obvious partitioning of interest would be to divide the sample space into positive-support elements (in distribution - ID) and the zero support elements (out-of-distribution - OOD); the partitioning is then $\mathcal{P}^{od} = \{A_o, A_+\}$; where $\{x \in A_+; p_x > 0,\ x \in \Omega\}$, $\{x \in A_o; p_x = 0,\ x \in \Omega\}$.

As our focus is on sample quality, we compare the ground truth distribution p to the empirical distribution $\hat{p}_\theta$ constructed from the samples of a generative model. For the synthetic experiments where we have access to p, we report:

- $Hel(p, \hat{p}_\theta)$ and $d_{TV}(p, \hat{p}_\theta)$,
- $d_{TV+} \triangleq \frac{1}{2}\sum_{x\in A_+}|p_x - \hat{p}_{\theta x}|$, $d_{TVo} \triangleq \frac{1}{2}\sum_{x\in A_o}|p_x - \hat{p}_{\theta x}|$,
- $\hat{p}_\theta(A_+) = \sum_{x\in A_+}\hat{p}_{\theta x}$; prob. estimates of valid sequences,
- $\hat{p}_\theta(A_i) = \sum_{x\in A_i}\hat{p}_{\theta x}$; prob. estimates of specified A_i.

Without ground truth distribution. In practice, p is not available. We still focus on generating samples that are representative of the distribution by comparing statistics of the ground truth distribution with those derived from generated samples. A major capability of interest of a generative model is its ability to properly capture patterns in the data; as such we can compare the higher order covariation of patterns of a generated set of samples to that of a test set. Such evaluation metrics are commonly used in the generative protein sequence modeling literature (Trinquier et al. 2021; McGee et al. 2021). Given a pattern of size p, described by positions and corresponding categories $(\{s_1, \ldots, s_p\}, \{k_1, \ldots, k_p\})$, and a set of M sequences $\mathbf{x}^M$, the higher order pattern covariation $C^{s_1,\ldots,s_p}_{k_1,\ldots,k_p}(\mathbf{x}^M)$ is the frequency of the appearance of the pattern in $\mathbf{x}^M$ minus the product of the frequencies of each individual element of the pattern:

$$\hat{f}^{s_1,\ldots,s_p}_{k_1,\ldots,k_p}(\mathbf{x}^M) = \frac{1}{M}\sum_{i=1}^M \mathbb{1}[x^i_{(s_1)} = k_1, \ldots, x^i_{(s_p)} = k_p],$$

$$C^{s_1,\ldots,s_p}_{k_1,\ldots,k_p}(\mathbf{x}^M) = \hat{f}^{s_1,\ldots,s_p}_{k_1,\ldots,k_p}(\mathbf{x}^M) - \prod_{j=1}^p \hat{f}^{s_j}_{k_j}(\mathbf{x}^M). \tag{11}$$

For a given pattern length p, we select a random subset of all possible patterns $\{pattern^p_1, \ldots\}$ by following the procedure described in (McGee et al. 2021), which focuses on the most likely patterns (the detailed selection procedure is described in the supplementary). We

		Hel	d_{TV}	$(d_{TV+}$	$d_{TVood})$	$p(A_{likely})$	$p(A_{rare})$	$p(A_+)$
$\mathbf{K=6}$	CNF+	37.02	34.63	25.59	9.04	40.91	41.01	81.92
	argmaxAR+	24.22	17.90	13.55	4.35	66.32	24.97	91.29
	CDM	19.27	16.19	14.34	1.84	66.33	29.98	96.31
	GMCD	**16.62**$*$	**16.07**	15.56	0.51	**75.71**$*$	**23.27**$*$	**98.98**$*$
$\mathbf{K=8}$	CNF+	81.86	83.63	69.79	13.84	35.98	36.34	72.32
	argmaxAR+	73.48	76.06	73.19	2.87	66.94	27.32	94.27
	CDM	73.65	76.36	73.81	2.55	62.70	32.20	94.90
	GMCD	**72.30**$*$	**74.98**$*$	73.34	1.64	**71.77**$*$	**24.94**$*$	**96.71**$*$
$\mathbf{K=10}$	CNF+	98.28	99.81	83.43	16.38	33.64	33.59	67.23
	argmaxAR+	98.43	99.81	76.95	22.86	40.38	13.89	54.27
	CDM	97.32	99.69	97.39	2.30	64.62	30.77	95.40
	GMCD	**97.27**$*$	**99.68**$*$	97.71	1.98	**66.99**$*$	**29.05**$*$	**96.05**$*$
	optimal	0	0			75	25	100

Table 1: Distances metrics and probability estimates for partitionings $\mathcal{P}, \mathcal{P}^{od}$ for the synthetic datasets.$*$ indicates significance w.r.t. to the Wilcoxon signed-rank test at the 5% level. $+$ indicates that more epochs were required to reach competitive results.

report the Pearson correlation ρ^p between the pattern higher order covariations computed on the test set $\mathbf{C}_p = [C^{pattern_1^p}(\mathbf{x}), \dots], \mathbf{x} \sim \mathcal{D}$ and the set of generated samples $\mathbf{C}_{\hat{p}_\theta} = [C^{pattern_1^p}(\mathbf{x}), \dots], \mathbf{x} \sim \hat{p}_\theta$.

Datasets

We design a ground truth distribution to generate a synthetic dataset of sequences of length $S = K$. We define the sample space $\Omega^K = \mathcal{C}^K$ and only assign probability mass on permutations of $\mathcal{C}$, i.e., $A_+ = \{\mathbf{x}; x_{(i)} \neq x_{(j)}\ \forall i \neq j\}$. Finally, we separate the positive sets in two and assign 3 times more mass to sequences with a "smaller" category at the start of the sequence than at the end, i.e.:

$$p(\mathbf{x}) = \begin{cases} \frac{3}{2K!} & \text{if } \mathbf{x} \in A_{likely} = \{\mathbf{x}; \mathbf{x} \in A_+ \wedge x_{(1)} < x_{(S)}\}, \\ \frac{1}{2K!} & \text{if } \mathbf{x} \in A_{rare} = \{\mathbf{x}; \mathbf{x} \in A_+ \wedge x_{(1)} > x_{(S)}\}, \\ 0 & \text{otherwise .} \end{cases}$$

This synthetic dataset is designed to emulate characteristics of a real world dataset. In practice, the distributions that we wish to model are likely to have positive support on a very small fraction of the probability space. Whether we are trying to generate text, images or proteins, the likelihood of stumbling across a "valid" sample when drawing from a uniform distribution is extremely small.

Natural partitionings of interest for this type of dataset are: 1) $\mathcal{P}^{od}$ as previously described where we can see a model's ability to grasp the positive support of the sample space; and 2) $\mathcal{P} = \{A_{likely}, A_{rare}, A_o\}$ where we can see a model's ability to assign the right amount of probability mass to the different sets.

	argmaxAR+	CNF+	CDM	GMCD
num. params	$250K$	$180K$	**40K**	**40K**
epoch time	1.9x	1.6x	**1x**	**1x**
sampling time	1.2x	1.2x	1.1x	**1x**

Table 2: Timing with $K = 8$. Experiments are conducted on GPU machines NVIDIA GeForce RTX 2060 .

We consider a small scale experiment $K = 6$ where the models are exposed to the entire ID set A_+ multiple times, a medium scale experiment $K = 8$ where the models are exposed to a sizeable fraction of the ID set, and a larger scale experiment $K = 10$ where the models are exposed to less than 1% of A_+ (see Table **??** for additional details).

As a real world application, we measure the performance of the models on two protein datasets from the Pfam protein family : **PF00076**, which contains $N = 137,605$ proteins of length $S = 70$ and **PF00014**, which contains $N = 13,600$ proteins of length $S = 53$. The number of categories for both datasets corresponds to the list of amino acids $K = 21$.

Experiment Details

Baselines. We compare our GMCD approach to three state-of-the-art baselines; 1) CNF (Lippe and Gavves 2021), a normalizing flow method that learns a mapping to/from the categorical space; 2) CDM (Hoogeboom et al. 2021a), a diffusion-based model; and 3) argmaxAR (Hoogeboom et al. 2021b), a normalizing flow method that uses an argmax operation to map to the discrete space. We select the autoregressive version because it was reported as the best alternative.

Experimental set-up. We train all models using the RAdam optimizer (Liu et al. 2020) and early stopping and keep the best model evaluated on the validation set. For the proteins dataset and for the large scale synthetic experiment $K = 10$, in order to avoid overfitting, we monitor to ensure that the model is not reproducing more than 1% of the training dataset in its generated samples. Performance metrics are averaged over 10 trials of $M = 10,000$ generated samples. A split of 70/20/10 is used for the protein datasets. The $p_\theta(\mathbf{X}|\mathbf{Z}^t, t)$ function is modeled using a non-autoregressive transformer similar to that used in (Hoogeboom et al. 2021a). Following (Ho, Jain, and Abbeel 2020), we use sinusoidal position embedding to process the time step t and concatenate it to $\mathbf{Z}^t$ to form the input to the transformer. The means $\boldsymbol{\mu}_1^*, \dots$ are computed using the procedure from (Gamal et al. 1987) , which employs simulated

		ρ^2	ρ^3	ρ^4	ρ^5	ρ^6	ρ^7	ρ^8	ρ^9
PF00076	CNF	-	-	-	-	-	-	-	-
	argmaxAR	73.13	73.00	69.85	63.88	58.74	49.08	49.74	56.03
	CDM	82.30	82.44	80.48	78.08	74.95	73.10	75.56	77.27
	GMCD	**84.19**$*$	**82.85**	**82.36**$*$	**81.04**$*$	**77.67**$*$	**78.09**$*$	**78.95**$*$	**80.39**$*$
PF00014	CNF	-	-	-	-	-	-	-	-
	argmaxAR	78.06	79.05	80.89	83.57	84.97	**88.51**	91.26	91.46
	CDM	**81.28**$*$	80.48	78.98	78.76	77.35	80.40	85.31	89.90
	GMCD	80.41	**80.81**$*$	**82.01**$*$	**84.09**$*$	**85.83**	88.39	**91.50**	**93.04**
abl.	GMCD random	69.21	69.04	71.69	78.81	80.26	87.65	90.54	93.20
	GMCD sharp	79.86	79.13	81.07	82.96	83.85	85.26	88.78	89.80

Table 3: Proteins experiment results. $-$ indicates that the pearson coefficient was not significant at the 5% level.

		ρ^2	ρ^3	ρ^4	ρ^5
K = 6	argmaxAR+	63.16	58.38	59.22	63.66
	CNF+	-12.97	14.61	-5.05	-21.10
	CDM	54.00	58.26	59.07	63.52
	GMCD	**64.03**	**63.88**	**66.22**	**67.64**
K = 8	argmaxAR+	30.93	21.49	13.90	**14.24**
	CNF+	-10.13	$-$	3.20	-1.31
	CDM	20.19	16.61	12.03	6.98
	GMCD	**32.70**	**26.83**$*$	**16.31**	10.71
K = 10	argmaxAR+	11.85	6.54	4.57	1.55
	CNF+	$-$	-4.68	-1.62	$-$
	CDM	22.46	13.40	**7.80**	4.55
	GMCD	**25.19**	**18.67**	6.60	**4.79**

Table 4: Pattern covariance metrics.

annealing. We provide a complete description of architectures, the hyperparameters selection procedure in the supplementary. The source code is available at https://github.com/networkslab/gmcd.

Results. Experiments on the synthetic dataset highlight the modeling capability of GMCD. For every scale that we consider, $K = 6, 8, 10$, GMCD outperforms at every distribution granularity: $\Omega, \mathcal{P}, \mathcal{P}^{od}$ (Table 1). This is reflected in the covariance pattern metrics (Table 4). The decomposition of d_{TV} into the two regions d_{TV+} and d_{TVo} shows that most of the error for all baselines comes from d_{TV+}, which is the error in estimating the probability mass of the valid sequences in A_+. This is to be expected as it is a harder task. CDM is the closest competitor and its generated samples are almost all valid ($p(A_+)$ is close to 100). Its deficiency is in assigning a probability mass ratio of approximately 2:1 to the two sets A_{likely} and A_{rare}. This results in higher statistical distance metrics d_{TV} and Hel. argmaxAR struggles to identify A_+ and requires additional training to reach a competitive result, but given more training time it can assign slightly better mass to the two sets, except for the larger scale experiment $K = 10$. CNF is unable to distinguish between the likely and rare sets, which greatly impedes its performance for all metrics. As expected, as the problem grows harder, the fine-grained metrics d_{TV}, Hel cannot be meaningfully estimated with this sample size. For the protein dataset, GMCD is the best method overall and performs consistently for every pattern size (Table 3).

S, K	$\lvert\Omega\rvert$	$\lvert A_+\rvert$	% A_+ in training set
6	$6^6 = 46,656$	$6! = 720$	100%
8	$8^8 = 16,777,2164$	$8! = 40,320$	21.34%
10	10^{10}	$3,628,800$	0.28%

Table 5: Size of the sample space $|\Omega|$, of the positive support set $|A_+|$(number of valid sequences) and the fraction of valid sequence contained in the training set of the synthetic datasets. We generate 10K sequences for the train/valid/test set for a total size $N = 30K$.

Ablation study and Time Analysis. We report ablation studies to verify the relative contribution of two model components. We compare with a GMCD version with no sphere packing algorithm. The category distributions are randomly placed with no optimization (GMCD random). We also report GMCD trained with the initial sequence as in (9) (GMCD sharp). This eliminates data augmentation. As shown in the bottom of Table 3 the ablation experiment conducted on the PF00014 datasets confirms the relative importance of the components. We also include time and memory complexity of training and sampling of the models in the **abl.** section of Table 2. GMCD requires the least time both for training and sampling because we can reduce the number of steps in the diffusion due to the more structured denoising procedure.

Conclusion

In conclusion, we introduced the GMCD model; a continuous diffusion-based model for nominal data. We introduced a novel novel fixed encoding procedure to map categorical data to the continuous space and gain representation flexibility. This also leads to a novel continuous denoising process that is cognizant of the categorical nature of the targeted distribution. The GMCD is fast to train, fast to sample from and generates representative samples of the ground truth distribution as demonstrated on synthetic and on a real world datasets.

References

Bond-Taylor, S.; Leach, A.; Long, Y.; and Willcocks, C. 2022. Deep Generative Modelling: A Comparative Review of VAEs, GANs, Normalizing Flows, Energy-Based and Autoregressive Models. *IEEE Trans. Patt. Analysis Machine Intelligence*.

Borji, A. 2019. Pros and cons of GAN evaluation measures. *Computer Vision and Image Understanding*, 179: 41–65.

Brookes, D.; Park, H.; and Listgarten, J. 2019. Conditioning by adaptive sampling for robust design. In *Proc. Int. Conf. Machine Learning ICML*, 773–782.

Caccia, M.; Caccia, L.; Fedus, W.; Larochelle, H.; Pineau, J.; and Charlin, L. 2020. Language GANs Falling Short. In *Proc. Int. Conf. Learning Representations ICLR*.

Celikyilmaz, A.; Clark, E.; and Gao, J. 2020. Evaluation of Text Generation: A Survey. ArXiv preprint: arXiv 2006.14799.

Child, R.; Gray, S.; Radford, A.; and Sutskever, I. 2019. Generating Long Sequences with Sparse Transformers. *CoRR*, abs/1904.10509.

Cooijmans, T.; Ballas, N.; Laurent, C.; Gülçehre, Ç.; and Courville, A. 2017. Recurrent Batch Normalization. In *Proc. Int. Conf. Learning Representations ICLR*.

Dai, Z.; Yang, Z.; Yang, Y.; Carbonell, J.; Le, Q.; and Salakhutdinov, R. 2019. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In *Proc. of the 57th Conference of the Association for Computational Linguistics, ACL*, 2978–2988.

Dinh, L.; Sohl-Dickstein, J.; and Bengio, S. 2017. Density estimation using Real NVP. In *Proc. Int. Conf. Learning Representations, ICLR*.

El-Gebali, S.; Mistry, J.; Bateman, A.; Eddy, S. R.; Luciani, A.; Potter, S. C.; Qureshi, M.; Richardson, L. J.; Salazar, G. A.; Smart, A.; Sonnhammer, E. L. L.; Hirsh, L.; Paladin, L.; Piovesan, D.; Tosatto, S. C. E.; and Finn, R. D. 2019. The Pfam protein families database in 2019. *Nucleic Acids Res.*, 47(D1): D427–D432.

Gamal, A.; Hemachandra, L.; Shperling, I.; and Wei, V. 1987. Using simulated annealing to design good codes. *IEEE Trans. on Info. Theory*, 33(1): 116–123.

Garbacea, C.; Carton, S.; Yan, S.; and Mei, Q. 2019. Judge the Judges: A Large-Scale Evaluation Study of Neural Language Models for Online Review Generation. In *Proc. Conf. on Empirical Methods in Natural Language Process. and Int. Joint Conf.e on Natural Language Process. (EMNLP-IJCNLP)*.

Hershey, J. R.; and Olsen, P. A. 2007. Approximating the Kullback Leibler Divergence Between Gaussian Mixture Models. In *Proc. 2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, volume 4, IV–317–IV–320.

Ho, J.; Chen, X.; Srinivas, A.; Duan, Y.; and Abbeel, P. 2019. Flow++: Improving Flow-Based Generative Models with Variational Dequantization and Architecture Design. In *Proc. Int. Conf. Machine Learning ICML*.

Ho, J.; Jain, A.; and Abbeel, P. 2020. Denoising Diffusion Probabilistic Models. In *Proc. Adv. Neural Info. Process. Syst. NeurIPS*.

Hoffman, S. C.; Chenthamarakshan, V.; Wadhawan, K.; Chen, P.-Y.; and Das, P. 2022. Optimizing molecules using efficient queries from property evaluations. *Nature Machine Intelligence*, 4(1): 21–31.

Hoogeboom, E.; Nielsen, D.; Jaini, P.; Forré, P.; and Welling, M. 2021a. Argmax Flows and Multinomial Diffusion: Learning Categorical Distributions. In *Proc. Adv. Neural Info. Process. Syst. NeurIPS*.

Hoogeboom, E.; Nielsen, D.; Jaini, P.; Forré, P.; and Welling, M. 2021b. Argmax Flows: Learning Categorical Distributions with Normalizing Flows. In *Proc. Symposium on Adv. in Appr. Bayesian Inference*.

Hua, W.; Dai, Z.; Liu, H.; and Le, Q. 2022. Transformer Quality in Linear Time. In *Proc. Int. Conf. Machine Learning ICML*, 9099–9117.

Jain, M.; Bengio, E.; Hernandez-Garcia, A.; Rector-Brooks, J.; Dossou, B. F. P.; Ekbote, C. A.; Fu, J.; Zhang, T.; Kilgour, M.; Zhang, D.; Simine, L.; Das, P.; and Bengio, Y. 2022. Biological Sequence Design with GFlowNets. In *Proc. Int. Conf. Machine Learning ICML*, 9786–9801.

Jun, H.; Child, R.; Chen, M.; Schulman, J.; Ramesh, A.; Radford, A.; and Sutskever, I. 2020. Distribution Augmentation for Generative Modeling. In *Proc. Int. Conf. Machine Learning ICML*, 5006–5019.

Katharopoulos, A.; Vyas, A.; Pappas, N.; and Fleuret, F. 2020. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention. In *Proc. Int. Conf. Machine Learning ICML*.

Kingma, D. P.; Salimans, T.; Poole, B.; and Ho, J. 2021. Variational Diffusion Models. In *Proc. Adv. Neural Info. Process. Syst. NeurIPS*.

Kitaev, N.; Kaiser, L.; and Levskaya, A. 2020. Reformer: The Efficient Transformer. In *Proc. Int. Conf. Learning Representations ICLR*.

Kumar, A.; and Levine, S. 2020. Model Inversion Networks for Model-Based Optimization. In *Proc. Adv. Neural Info. Process. Syst. NeurIPS*, 5126–5137.

Lin, T.; Wang, Y.; Liu, X.; and Qiu, X. 2021. A Survey of Transformers. *arXiv e-prints*, arXiv:2106.04554.

Lippe, P.; and Gavves, E. 2021. Categorical Normalizing Flows via Continuous Transformations. In *Proc. Int. Conf. Learning Representations ICLR*.

Liu, L.; Jiang, H.; He, P.; Chen, W.; Liu, X.; Gao, J.; and Han, J. 2020. On the Variance of the Adaptive Learning Rate and Beyond. In *Proc. Int. Conf. Learning Representations ICLR*.

Lucic, M.; Kurach, K.; Michalski, M.; Bousquet, O.; and Gelly, S. 2018. Are GANs Created Equal? A Large-Scale Study. In *Proc. Adv. Neural Info. Process. Syst. NeurIPS*.

McGee, F.; Hauri, S.; Novinger, Q.; Vucetic, S.; Levy, R.; Carnevale, V.; and Haldane, A. 2021. The generative capacity of probabilistic protein sequence models. *Nature Communications*, 12.

Nagarajan, V.; Andreassen, A.; and Neyshabur, B. 2021. Understanding the failure modes of out-of-distribution generalization. In *Proc. Int. Conf. Learning Representations ICLR*.

Nichol, A. Q.; and Dhariwal, P. 2021. Improved Denoising Diffusion Probabilistic Models. In *Proc. Int. Conf. Machine Learning ICML*.

Rabanser, S.; Günnemann, S.; and Lipton, Z. 2019. Failing Loudly: An Empirical Study of Methods for Detecting Dataset Shift. In *Proc. Adv. Neural Info. Process. Syst. NeurIPS*.

Salimans, T.; and Ho, J. 2022. Progressive Distillation for Fast Sampling of Diffusion Models. In *International Conference on Learning Representations*.

Sinha, A.; Song, J.; Meng, C.; and Ermon, S. 2021. D2C: Diffusion-Decoding Models for Few-Shot Conditional Generation. In Ranzato, M.; Beygelzimer, A.; Dauphin, Y.; Liang, P.; and Vaughan, J. W., eds., *Proc. Adv. Neural Info. Process. Syst. NeurIPS*, volume 34, 12533–12548. Curran Associates, Inc.

Socolich, M.; Lockless, S. W.; Russ, W. P.; Lee, H.; Gardner, K. H.; and Ranganathan, R. 2005. Evolutionary information for specifying a protein fold. *Nature*, 437(7058): 512–518.

Sohl-Dickstein, J.; Weiss, E.; Maheswaranathan, N.; and Ganguli, S. 2015. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. In *Proc. Int. Conf. Machine Learning ICML*.

Theis, L.; van den Oord, A.; and Bethge, M. 2016. A note on the evaluation of generative models. In *Proc. Int. Conf. Learning Representations ICLR*.

Thompson, R.; Knyazev, B.; Ghalebi, E.; Kim, J.; and Taylor, G. W. 2022. On Evaluation Metrics for Graph Generative Models. In *Proc. Int. Conf. Learning Representations ICLR*.

Trinquier, J.; Uguzzoni, G.; Pagnani, A.; Zamponi, F.; and Weigt, M. 2021. Efficient generative modeling of protein sequences using simple autoregressive models. *Nature Communications*, 12.

Tubiana, J.; Cocco, S.; and Monasson, R. 2019. Learning Compositional Representations of Interacting Systems with Restricted Boltzmann Machines: Comparative Study of Lattice Proteins. *Neural Comput.*, 31(8): 1671–1717.

Uria, B.; Murray, I.; and Larochelle, H. 2013. RNADE: The Real-Valued Neural Autoregressive Density-Estimator. In *Proc. Adv. Neural Info. Process. Syst. NeurIPS*.

Vahdat, A.; Kreis, K.; and Kautz, J. 2021. Score-based Generative Modeling in Latent Space. In *Proc. Adv. Neural Info. Process. Syst. NeurIPS*.

Wu, Y.; Burda, Y.; Salakhutdinov, R.; and Grosse, R. B. 2017. On the Quantitative Analysis of Decoder-Based Generative Models. ArXiv preprint: arXiv 1611.04273.

Xiao, Z.; Kreis, K.; and Vahdat, A. 2022. Tackling the Generative Learning Trilemma with Denoising Diffusion GANs. In *Proc. Int. Conf. Learning Representations ICLR*.

Zhou, S.; Gordon, M. L.; Krishna, R.; Narcomey, A.; Fei-Fei, L.; and Bernstein, M. S. 2019. HYPE: A Benchmark for Human eYe Perceptual Evaluation of Generative Models. In *Proc. Adv. Neural Info. Process. Syst. NeurIPS*.

Ziegler, Z.; and Rush, A. 2019. Latent Normalizing Flows for Discrete Sequences. In *Proc. Int. Conf. Machine Learning ICML*.