MLM Diffusion: Generating Globally-Consistent High-Resolution Images from Discrete Latent Spaces

PETER HESSEY

MLM Diffusion: Generating Globally-Consistent High-Resolution Images from Discrete Latent Spaces

Student Name: Peter Hessey

Supervisor Name: C. G. Willcocks

Submitted as part of the degree of MScR Computer Science to the

Board of Examiners in the Department of Computer Sciences, Durham University

15/03/22

Abstract —

Context/Background: Creating deep generative models capable of generating high-resolution images is a critical challenge for modern deep learning research, with far-reaching impacts in domains such as medical imaging and computer graphics. One method that has recently achieved great success in tackling this problem is probabilistic denoising diffusion. However, whilst diffusion models can generate high quality image content, key limitations remain in terms of high computational requirements.

Aims: This thesis investigates new techniques to overcome the computational cost requirements that currently limit generative diffusion models. Specifically, this thesis focuses on training deep learning models to model and sample from discrete latent spaces that can be used to generate high-resolution images.

Method: This thesis introduces a novel type of diffusion probabilistic model prior capable of generating discrete latent representations of high-resolution images by utilising bidirectional transformers. The quality and diversity of images generated by these models are then evaluated and compared quantitatively and qualitatively to other similar models, before other interesting properties are also explored.

Results: The proposed approach achieves state-of-the-art results in terms of Density (LSUN Bedroom: 1.51; LSUN Churches: 1.12; FFHQ: 1.20) and Coverage (LSUN Bedroom: 0.83; LSUN Churches: 0.73; FFHQ: 0.80), and performs competitively on FID (LSUN Bedroom: 3.64; LSUN Churches: 4.07; FFHQ: 6.11) whilst also offering significant advantages in terms of computation time.

Conclusions: Through the use of powerful bidirectional transformers and discretised latent spaces, it is possible to train a discrete diffusion model to generate high-quality, high-resolution images in only a fraction of the time required by continuous diffusion probabilistic models trained on the data space. Not only are these models faster to train and sample from, they also only require a single NVIDIA 2080ti GPU with 11GB of RAM for successful training and achieve state-of-the-art results in terms of generated image quality and diversity.

Keywords — Deep Learning, Generative Models, Denoising Diffusion, Discrete Latent Spaces

Contents

Declaration

The majority of the experiments detailed in this thesis were carried out for the completion of a paper submitted to ECCV 2022, which, with equal contributions, was co-authored with a Durham University PhD researcher, Sam Bond-Taylor [3]. Further contributions were also made by Hiroshi Sasaki, Dr Chris G. Willcocks, and Professor Toby Breckon.

Acknowledgements

Having undertaken the entirety of my research Master's remotely due to COVID restrictions, it was often very difficult to find the motivation to keep working, especially when it felt like everything was going wrong and that I had no good ideas to pursue. However, my supervisor Dr. Chris Willcocks seems to have sixth sense for such things; he always knew exactly the right things to say in our weekly meetings to give me the inspiration I needed for the week ahead. He was a constant support for me throughout what was an incredibly difficult year, and for that I cannot thank him enough.

Alongside Chris I wish to thank his PhD student, Sam Bond-Taylor. His passion for all things deep learning, his outstanding ability to make the most complex topics seem trivial and easy-to-understand, as well as his patience in collaborating with an inexperienced Master's student such as myself, meant that I learned so much more this year than I ever imagined I would. Thank you Sam, it was a pleasure and honour to work with you and I sincerely hope we can do so again in the future.

I also would like to thank Dr Robert Powell and all the individuals who maintain and support Durham's NVIDIA CUDA Centre (NCC). Rob was always quick to help with any technical troubles and the NCC was a critical resource for all the work we carried out throughout the year.

Finally I want to thank my partner Gemma for supporting me constantly, as well as my friends and housemates Pétur and Ecem for putting up with my overly-enthusiastic computer science babbling.

Introduction

3.1 High-Resolution Image Generation

In the domains of deep learning and computer vision, the ability to generate new, high-resolution, photo-realistic imagery has long been a primary goal. For a model to be able to generate images in this way demonstrates that the target data space is modelled precisely and can thus be utilised for many useful downstream tasks. Recent advancements towards these goals have yielded direct benefits for fields such as medical image synthesis [18], computer graphics [9, 77], image editing [41], image-to-image translation [62], and image super-resolution [25].

As impressive as these modern advancements are, there still exist many monumental problems that must be overcome before high-resolution images can be generated efficiently and consistently. Current deep generative methods can in general be divided into five main classes [4], each of which have their own merits and flaws, as well as their own variety of techniques and trade-offs to scale to higher resolutions. The classes most relevant to this thesis are discussed briefly below, and in more detail in Sec. 4.

3.2 Existing Techniques

The ever-popular Generative Adversarial Networks (GANs) [19] achieve visually-impressive results and often dominate much of deep generative model research. Techniques for scaling GANs to high-resolutions include progressive growing [34], large batches [6], and regularisation [45, 42]. However, GANs are known to be difficult and unstable to train and lack the statistical foundations to enable them to estimate likelihood, a key requirement for many downstream tasks.

A family of powerful generative models that are well-founded in statistical theory is Variational Autoencoders (VAEs) [37]. These models can be scaled by building complex priors [10, 70, 73] and correcting the learned density [76]. Unfortunately, these scaling techniques come with severe computational costs and the constraints placed on VAE priors often limit their ability to be competitive with other generative models in terms of generated image quality.

Autoregressive approaches are a modern suite of powerful and flexible generative techniques that have demonstrated impressive results in the domain of image generation [72, 61]. Straightforward to train, these models directly optimise the likelihood of the target data space and can be scaled to higher resolutions by making independence assumptions [56] or partitioning spatial dimensions [43]. Autoregressive approaches have recently taken advantage of the popular transformer architecture [74] which is able to model long distance relationships using a powerful attention mechanism that can be trained in parallel. By constraining a transformer architecture to attend a fixed ordering of tokens in a unidirectional manner, it can be used to parameterise an autoregressive model for generative modelling [11, 50]. While effective and conceptually simple, autoregressive techniques still fail to scale to high-resolutions of images without incurring prohibitively-high computational costs and are prone to overfitting to training data unless carefully managed. Furthermore, enforcing a fixed ordering on the generated tokens is incongruous with the natural structure of image data and thereby limits the representation ability of the transformer and unnecessarily restricts the sampling process to be both sequential and slow.

Finally, and of particular relevance to this thesis, is the class of generative model class known as Energy Based Models (EBMs), specifically the subclass of EBMs known as Denoising Diffusion Probabilistic Models (DDPMs) [24, 64]. DDPMs convert the data space to a simple prior, such as a Gaussian distribution, through the use of a large number of very small changes that they subsequently learn to reverse, allowing them to model complex, high-dimensional data spaces using simple latent spaces that can be easily sampled from. While techniques do exist for scaling DDPMs to high-dimensional data spaces, such as Stochastic Differential Equations (SDEs) [66]

and cascades [25], they suffer from very slow sampling and training times.

3.3 Overview

In an attempt to improve upon existing techniques for high-resolution image generation, the primary contributions of this thesis are:

- The proposal of a novel parallel token prediction approach for generating discrete latent image representations, **MLM Diffusion**. This method is able to train on a single NVIDIA 2080ti GPU in only a few days and offers faster sampling times than existing autoregressive and DDPM approaches.
- State-of-the art image generation quality across three benchmark datasets in terms of Density (LSUN Bedroom: 1.51; LSUN Churches: 1.12; FFHQ: 1.20) and Coverage (LSUN Bedroom: 0.83; LSUN Churches: 0.73; FFHQ: 0.80), while also being competitive on FID (LSUN Bedroom: 3.64; LSUN Churches: 4.07; FFHQ: 6.11) and preserving the ability to estimate likelihood.
- Demonstration of the fact that models trained using this technique are able to generate globally consistent images at resolutions exceeding that of the original training data by aggregating multiple context windows, allowing for much larger context regions.

The related work section, Sec. 4, covers in detail the background knowledge necessary to understand the key components of the presented method. Sec. 5 explains the MLM Diffusion method in detail as well as discussing the training setup used. Results of all the experiments carried out for this thesis are then given in Sec. 6, including examples of generated images. Finally, a discussion of the models limitations can be found in Sec. 7, with conclusions and potential future work detailed in Sec. 8.

Figure 1: Super-resolution LSUN Bedrooms samples generated using MLM Diffusion models.

Related Work

The research carried out for this thesis draws upon ideas and depends upon model architectures from a broad range of deep learning research areas. Each of the architectures described in this section are currently being actively researched at the time of writing, and each comes with its own complexities, nuances and numerous variants.

This section aims to cover the fundamentals of all architectures and ideas used for the completion of this thesis, as well as explaining any of the necessary complexities required. For deeper explanations of the architectures used, please refer to the bibliography.

4.1 Discretised Generative Autoencoders

Generative autoencoder models are a family of simple yet effective deep generative models capable of producing impressive results [10, 69]. They rely on the learning of latent variables - variables affecting the data distribution that cannot be directly observed. Latent variables can be used to represent the data distribution in a simpler, more compressed format that allows for easier manipulation and sampling of the data distribution [28].

An autoencoder is composed of two key components, an *encoder* and a *decoder*. Given an initial data distribution X and latent variable distribution $\mathcal{Z}$, an encoder $q : X \to \mathcal{Z}$ will take $\boldsymbol{x} \sim X$ as input, and return latent variables $\boldsymbol{z} \sim \mathcal{Z}$. The decoder $p : \mathcal{Z} \to X$ will subsequently take latent variables $\boldsymbol{z}$ as input and return an estimation of the original data, $\hat{\boldsymbol{x}}$. An autoencoder model trained in this way will successfully be able to produce compressed forms of the original data. However, without some way to sample the latent space, new data cannot be generated using only this method.

4.1.1 Variational Autoencoders (VAEs)

VAEs introduced a method of training autoencoders to perform as generative models [37]. By imposing a prior on the distribution of latent variables during training, one is able to sample latents from that distribution and thus able to generate new, unseen data by passing the sampled latents into the trained decoder.

To impose this prior, the Kullback–Leibler divergence (KLD) [38] between the generated latents and a Gaussian prior is minimised, giving rise to the following loss function:

$$\mathcal{L}(\theta, \phi, \boldsymbol{x}) = -D_{kl}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z})) + \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \ln p_\theta(\boldsymbol{x}|\boldsymbol{z}), \tag{1}$$

where θ and ϕ are the parameters of the decoder and encoder respectively, $p_\phi(z)$ is the prior distribution of the latent space and D_{KL} is defined as

$$D_{kl}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z})) = \sum_{\boldsymbol{x} \in \mathcal{X}} q_\phi(\boldsymbol{z}|\boldsymbol{x}) \log \left(\frac{q_\phi(\boldsymbol{z}|\boldsymbol{x})}{p_\theta(\boldsymbol{z})} \right). \tag{2}$$

For image-data generation, a commonly used prior is the multivariate Gaussian distribution with diagonal covariance. This is as it is allows for tractable computation of Eq. (2), and it is possible to sample from in closed form. In combination with convolutional architectures to extract suitable latents, VAEs trained in this way are capable of generating image-data simply by sampling latents from a Gaussian distribution and passing them into a trained decoder. See Fig. 2 for a diagram of such an architecture.

Unfortunately, these simple models struggle to scale to high-dimensional data such as large images. The imposition of a simple prior on the latent space makes training and sampling straightforward, but makes many assumptions about the nature of the latent space. This results

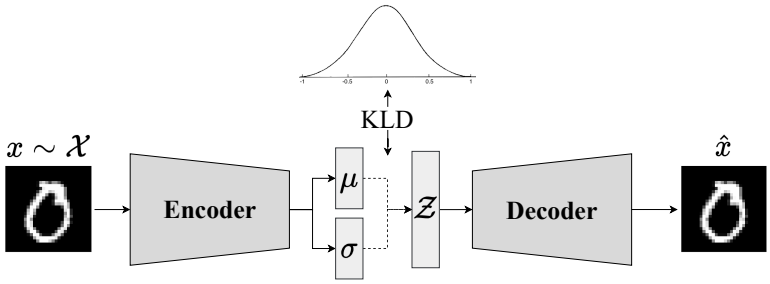


Figure 2: A visualisation of a variational autoencoder. The Kullback-Leibler Divergence between the latent space and a standard Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{1})$ is minimised, thus imposing a prior on the latent space.

in significant amounts of detail being lost and renders it impossible to faithfully reconstruct high-dimension input data, with image reconstructions often looking blurry. In order to scale to high-resolution images with more complex latent spaces, more sophisticated architectures are required [10].

4.1.2 Vector Quantised VAEs

In the above description of VAEs, it is assumed throughout that the latent variables are continuous values, as is the case in many latent-variable generative models. Vector Quantised VAEs (VQVAEs), do away with this assumption and instead use a specific type of autoencoder model that encodes input data into discretised latent variables, and reconstructs the original data from these discrete latents [73]. The primary benefit of using discrete latent variables is that by using a finite codebook of possible latent variables, each latent must contain a much richer set of information in order to reconstruct the input data faithfully. This enables VQVAEs to produce latents of lower dimensionality than their continuous counterparts, while still producing high-quality reconstructions [55].

Given a convolutional autoencoder architecture which downsamples to smaller spatial resolution, a simple approach to generate discrete latents is to use the argmax operation to map continuous latents to their closest elements in a finite codebook of vectors [73]. Specifically, for a codebook $\mathcal{E} \in \mathbb{R}^{K \times D}$, where K is the number of discrete codes in the codebook and D is the dimension of each code, latents are generated by first passing an input $\boldsymbol{x}$ through an encoder to produce continuous latents $E(\boldsymbol{x}) = \boldsymbol{z} = \{\boldsymbol{z}_1, \boldsymbol{z}_2, ..., \boldsymbol{z}_L\} \in \mathbb{R}^{L \times D}$, where L is the size of the latent space. Each $\boldsymbol{z}_i$ is then mapped via a nearest-neighbour lookup onto a discrete codebook value, $\boldsymbol{e}_j \in \mathcal{E}$ using the quantisation function $q(\boldsymbol{z})$:

$$q(\boldsymbol{z}) = \{\boldsymbol{q}_1, \boldsymbol{q}_2, ..., \boldsymbol{q}_L\} \text{ , where } \boldsymbol{q}_i = \min_{\boldsymbol{e}_j \in \mathcal{E}} ||\boldsymbol{q}_i - \boldsymbol{e}_j||. \tag{3}$$

In order to learn a codebook of discrete latents that can be used for accurate reconstructions, the individual codes are treated as parameters and are updated throughout training also. Therefore the resulting loss function to train a VQVAE can be defined as

$$\mathcal{L}_{VQ}(\theta, \phi, \boldsymbol{x}) = \log p_\theta(\boldsymbol{x}|q(\boldsymbol{z})) + ||\text{sg}[\boldsymbol{z}] - q(\boldsymbol{z})||_2^2 + \beta||\boldsymbol{z} - \text{sg}[q(\boldsymbol{z})]||_2^2, \tag{4}$$

where sg stands for the stopgradient operator that is defined as identity at forward computation time and has zero partial derivatives, thus effectively constraining its operand to be a non-updated constant [73]. The first term of this loss function specifies the reconstruction loss that updates the parameters of the encoder and decoder. The second term updates the values of

the codebook to move towards the encoder outputs, and the final term constraints the outputs of the encoder to prevent the volume of the embedding space (codebook) growing arbitrarily.

As there is no real gradient defined for the quantisation operation specified in equation Eq. (3), there are no gradients that can be directly backpropogated and used to train the encoder. In order to circumvent this problem, a straight-through gradient estimator [2] is used to copy the gradients from the decoder inputs onto the encoder outputs.

4.1.3 Codebook Collapse

A common problem in VQVAEs is *codebook collapse*. When codebook collapse occurs, a subset of the codebook is entirely ignored by the autoencoder and is never used, limiting the reconstruction potential of the VQVAE. A diversity loss as been proposed to mitigate this, however, this has been shown to result in worse sample quality [13]. An alternative quantisation approach that avoids this bias is the Gumbel-Softmax/Concrete distribution [54], a relaxed categorical distribution with a temperature value that controls the level of relaxation, but this comes at the risk of reduced reconstruction quality.

4.1.4 Sampling and Two-Stage Training

Unlike VAEs, VQVAEs do not have a pre-defined prior imposed on their latent space. This is as the continuous distributions such as Gaussian distributions would not be suitable for the discrete latents produced by VQVAEs and imposing discrete priors such as categorical distributions severely limits reconstruction and sample quality.

Therefore, VQVAEs are trained in a *two-stage training process*. In the first stage, the VQVAE is trained to reconstruct the target data, producing an information-rich codebook of discrete latents. The second training stage is where the prior is learned in order to be able to sample from this latent space, and is performed only once the first stage is complete. Learning to model a high-dimensional, discrete prior is a complex and challenging task and is the primary focus of this thesis. In the original work on VQVAEs, autoregressive models (explained in Sec. 4.3.1) are trained to model the prior.

4.1.5 Vector Quantised GANs

To improve the quality of VAE samples, introducing a hierarchy of latent variables where each layer is conditioned on the previous has been shown to be an effective approach for continuous VAEs [10]. While the same is true for VQVAEs [55], the second training stage requires multiple autoregressive models to be trained and consequently leads to very slow sampling.

To improve the representation capabilities of VQVAEs, Esser et. al. [17] proposed Vector Quantised Generative Adversarial Networks (VQGANs), that use an adversarial training approach [19] to generate discrete latent spaces.

In this case, adversarial training simply means that an additional model, known as a discriminator, is trained simultaneously alongside the autoencoder. The discriminator is trained to classify input data either as real or fake data, i.e. to identify if the input was sampled from the original dataset or if it is a reconstruction produced by the autoencoder. The autoencoder is then trained to try and 'trick' the discriminator into incorrectly classifying its reconstructions as real data resulting in the two models competing in an adversarial manner. This method allows the VQGAN to reconstruct high-resolution images faithfully without the use of complex hierarchical architectures.

4.1.6 Mode Collapse

'Mode collapse' is a term used in adversarial training to describe when the generator begins only generating a reduced subset of outputs that the discriminator fails to classify correctly. This is

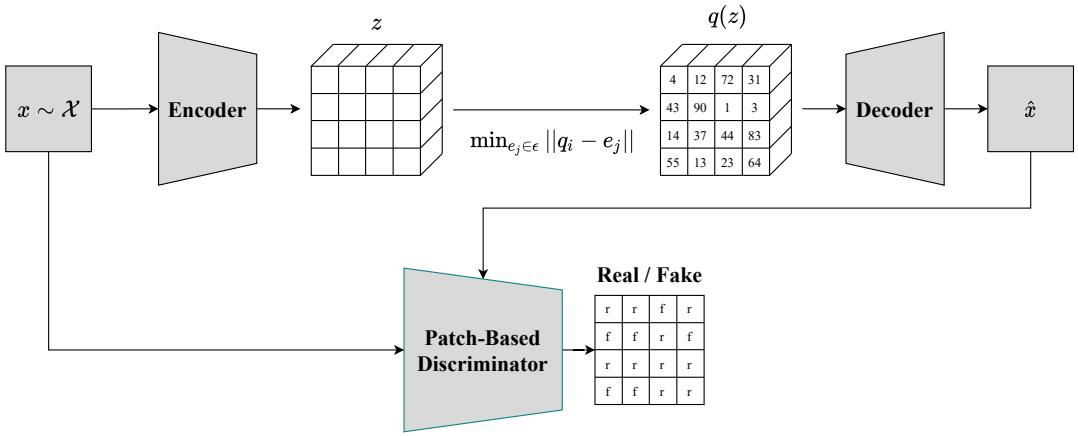


Figure 3: VQGAN Architecture [17]. Values in $q(\boldsymbol{z})$ are selected using Eq. (3) and correspond to values from a finite, discrete codebook.

a direct result of the instability inherent in adversarial training and is a phenomenon known to commonly occur in generative adversarial networks (GANs) [19], often resulting in generators failing to train poorly, if at all.

Fortunately, the mode collapse introduced by the patch-based discriminator [30] in the VQ-GAN model is much more moderate than that seen in generative models and results in the improvement of the high-resolution image reconstruction ability of the model. This is as slight mode collapse encourages the discrete latents to represent specific image components - such as a nose or a mouth for a dataset of human faces - and thereby enables the model to faithfully reconstruct input images using latent spaces that are smaller than the ones used by VQVAEs. These smaller latent spaces then result in reduced training time for the second stage when attempting to learn a prior over the latents, which is key to the sampling approaches introduced in Section 5.

4.2 EBMs

Energy-Based Models (EBMs) are a form of conceptually-simple and highly-flexible generative model that provide a framework for generating high-dimensional data across a range of domains [15]. EBMs rely on the observation that any probability density function can be expressed in the form

$$p(\boldsymbol{x}) = \frac{e^{-E(\boldsymbol{x})}}{\int_{\tilde{\boldsymbol{x}} \in X} e^{-E(\tilde{\boldsymbol{x}})}}, \tag{5}$$

where $E : \mathbb{R}^D \to \mathbb{R}$ is an energy function defined for data of dimension D. If a neural network is trained to accurately approximate the energy function of a given data distribution, then it becomes trivial to sample from the data distribution itself using the above equation.

However, the integral in Eq. (5) is intractable and requires approximating in order to train a model within a reasonable timeframe. The most commonly used and successful method for this is Contrastive Divergence [23]. Using this method, the energy values of samples from the data distribution are 'pushed' down, while samples taken from the generative model are 'pushed' up. By taking samples from the energy distribution using Monte Carlo Markov Chain (MCMC) approximation, the following loss function can be defined to train continuous, Implicit EBMs:

$$\nabla \mathcal{L}_\theta = \mathbb{E}_{\boldsymbol{x}^+ \sim p_d}[\nabla_\theta E_\theta(\boldsymbol{x}^+)] - \mathbb{E}_{\boldsymbol{x}^- \sim p_\theta}[\nabla_\theta E_\theta(\boldsymbol{x}^-)], \tag{6}$$

where p_d is the data distribution, p_θ is the distribution represented by the model with parameters θ, and $\boldsymbol{x}^+$ and $\boldsymbol{x}^-$ represent samples from the data and model distribution, respectively.

EBMs are very effective as generative models for a number of reasons. Firstly, unlike VAEs, they model the probability distribution of the data without the need for imposing priors, allowing them to accurately model all modalities. Furthermore, EBMs are straightforward to train and have a high level of generality, making them suitable for use as both generative and discriminative models [21]. Finally, data points in an EBM sample have access to and can be affected by all other data points in that sample. This is very important in the context of image-generation, as it allows for the generation of globally-consistent images, unlike autoregressive methods (Sec. 4.3.1).

While EBMs are very popular thanks to the aforementioned properties, they aren't without their downsides. Specifically, implicit EBMs are notoriously difficult to train, due to their slow training time, sampling time and instability when training using Contrastive Divergence. These factors make them poorly-suited for generative modelling of high-dimensional data such as high-resolution image data. However, most importantly for this thesis, EBMs demonstrate the possibility of generating samples using a global context without imposing simple priors, and were one of the primary inspirations for the methods introduced in this thesis.

4.2.1 Discrete EBMs

In recent years deep EBMs have been used as generative models almost exclusively on continuous data spaces [15, 76]. In such data spaces the gradients are readily available through differentiation of the energy function, E, and can be utilised to perform MCMC sampling. Such easily-computable gradients do not exist for discrete distributions such as categorical distributions, making MCMC sampling in discrete data spaces a far more complex task and very difficult to scale to high-dimensional data. Most existing approaches for speeding up discrete sampling require a priori information or assumptions about the structure and independence of the data. Unfortunately, such information is often not available (e.g. in the case of learned discrete latent spaces) and making any assumptions, such as imposing discrete priors, will destroy vital information in the data space.

Gibbs-With-Gradients Recent work in this domain has yielded a new method for sampling discrete data spaces with significantly-improved speed and effectiveness. This method, know as Gibbs-With-Gradients (GWG), takes advantage of the observation that many discrete distributions are implemented as differentiable functions of continuous inputs, specifically by restricting the continuous inputs to a subset of their domain [20]. GWG gradients utilises these underlying gradients to inform Metropolis-Hastings proposals [44] for updating the discrete samples in an MCMC chain.

For D-dimensional categorical data (like that used for discrete latent spaces), creating a new sample proposal requires first selecting which dimension i to change given an input x:

$$q(i|x) = \text{Categorical}\left(\text{softmax}\left(\frac{\tilde{d}(x)}{2}\right)\right), \tag{7}$$

where $\tilde{d}(x)$ estimates the likelihood ratios between a given input x and other possible states x' and is defined as

$$\tilde{d}(x)_{ij} = \nabla_x E(x)_{ij} - x_i^T \nabla_x E(x)_i, \tag{8}$$

where j is the new value to update dimension i to, and $E(x)$ is the energy function of the Discrete EBM.

Using Eq. (7) and Eq. (8) it is possible to produce discrete samples $x^- \sim p_\theta$ and therefore train a discrete EBM using contrastive divergence as in Eq. (6). These models preserve

all the benefits of EBMs such as global-consistency while also accurately modelling discrete distributions.

4.3 Transformers

4.3.1 Autoregressive Models

Autoregressive models are a family of powerful generative models capable of directly maximising the likelihood of the data on which they are trained. These models have achieved impressive image generation results in recent years but their sequential nature limits them to relatively low dimensional data [72, 71, 11, 32, 61, 59].

The training and inference process for autoregressive models is based on the chain rule of probability. Given a variable $\boldsymbol{x}$ that can be decomposed into individual components $\boldsymbol{x} = \{x_1, ..., x_n\}$, an autoregressive model with parameters θ generates new samples sequentially:

$$p_\theta(\boldsymbol{x}) = p_\theta(x_1, x_2, ..., x_n) = \prod_{i=1}^{n} p_\theta(x_i | x_1, ..., x_{i-1}). \tag{9}$$

A significant downside of this sequential process is the fact that the order in which components are generated has to be selected manually. While this is trivial for data with a predefined order (e.g. text, audio and time-series data), selecting an ordering for data such as images has a significant impact on training effectiveness and sample quality, with no clear optimal ordering.

Furthermore, regardless of the pixel ordering selected for generating images, each generated pixel only has access to the information available in pixels that have been already generated as part of the sequence. This results in the initial pixels in the sequence having access to very little information and presents a number of issues when generating images. This limitation in information-access causes a lack of global-consistency in generated images, as without access to all other pixels it is impossible to make new pixels globally-coherent.

Despite these issues, and the fact that the sequential training and sampling process is prohibitively slow for high-dimensional data such as high-resolution images, autoregressive models are very effective and commonly used for modelling lower-dimensional data. This makes autoregressive models a popular choice for modelling priors over discrete latents to generate new images as part of a two-stage training process [17, 73].

4.3.2 Attention & Self-Attention

Attention is a mechanism by which a machine learning model can selectively attend to a given set of inputs, allowing the model to make predictions and generate new data by utilising the relationships between data points in the input sequences [74]. For attention to work, weight matrices are learned that map how important each data point is to all other data points, and vice versa. This mechanism is the core component in a family of machine learning models known as 'transformers'. Transformers were initially used primarily for language tasks such as neural machine translation (NMT) and question-answer dialogue generation and they have recently been successfully utilised for a range of image-based tasks [50, 17].

A specific type of attention, known as self-attention, describes the approach where instead of computing attention between 2 or more input sequences, attention is instead calculated between an input sequence and itself. Using this method, it is possible to attend to a subset of a data sample in order to produce predictions about other points in the data. This thesis will focus exclusively on self-attention as opposed to other forms of attention, as it is the attention method best suited for generative modelling and is a key aspect of the methods introduced in Sec. 5

Scaled Dot-Product Attention There are many forms of self-attention, such as additive attention, general attention and dot-product attention. Each of these methods come with their pros, cons and specific use-cases. However, by far the most popular and successful self-attention mechanism is *Scaled Dot-Product Attention*.

Scaled Dot-Product Attention follows a number of key steps to compute attention on an input sequence. Firstly, inputs are encoded as key-value pairs, ($\boldsymbol{K}$, $\boldsymbol{V}$), where $\boldsymbol{V}$ represents the inputs and the keys $\boldsymbol{K}$ perform the role of indexing the inputs. During training a set of queries, $\boldsymbol{Q}$ is made. The dot-product of the keys and queries is then calculated, producing a similarity vector that describes which value vectors should be accessed [4]. This process is described using the following equation:

$$\text{Attn}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}}\right)\boldsymbol{V}, \tag{10}$$

where d_k is the dimension size of the query and key vectors and performs the role of normalising the gradient magnitudes in order to increase the learning efficiency of the transformer model when inputs are large.

Multi-Head Attention The final component that enables transformers to be so powerful and efficient to train is multi-head attention. This is not another form of attention but rather an approach to optimising attention weight matrices more efficiently. Rather than computing attention once for the entire input, multi-head attention instead linearly projects the entire input into multiple smaller chunks before than computing attention on all chunks in parallel (see Fig. 4). The attention outputs for each of these chunks are then concatenated before finally being passed into a linear layer to map back to the desired output dimension (e.g. the image size for generating images):

$$\begin{aligned} \text{MultiHeadAttention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) &= [\text{head}_1; ...; \text{head}_h]\boldsymbol{W}^O \\ \text{head}_i &= \text{Attention}(\boldsymbol{Q}\boldsymbol{W}_i^q, \boldsymbol{K}\boldsymbol{W}_i^k, \boldsymbol{V}\boldsymbol{W}_i^v) \end{aligned}, \tag{11}$$

where the weights matrices $\mathbf{W}$ represent the linear layer weights that project each matrix to the correct dimensions [74].

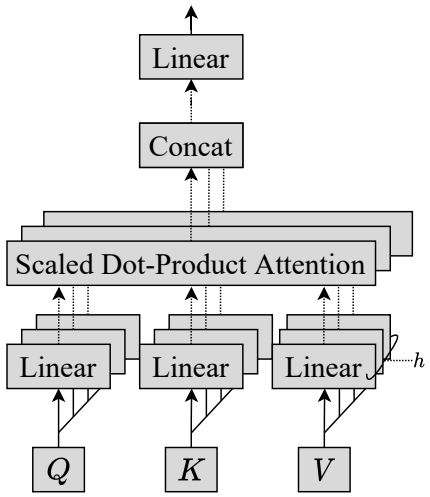


Figure 4: Visualisation of Multi-Head Attention block, where h is the number of heads. [74].

By utilising multi-head attention, transformers are able to calculate attention in parallel and thus train efficiently and stably. These models are usually implemented in an encoder-decoder structure, where the encoder attends to an input sequence to produce an attention-based representation. This representation can then be used to locate key information and relationships between data points using a large context. A decoder can then take this representation as input and subsequently autoregressively generate outputs. The decoder uses the same architecture as the encoder, but with an added block that masks future inputs to prevent the decoder from 'cheating' by attending to inputs it shouldn't have access to yet. This architectural restriction results in standard transformers being trained in an autoregressive manner.

While transformers are incredibly effective for language-based tasks thanks to their selective attention, high parallelisability and ease of implementation, they are not without their flaws, especially in the realm of image-generation. The primary issue with standard generative approaches using transformers is that they are autoregressive, which entails all of the image-generation problems listed in Sec. 4.3.1. Furthermore, the memory requirements of the softmax function in Eq. (10) scales quadratically with respect to the data dimension, meaning that hardware-restrictions make it impossible to compute global attention for high-resolution images. This results in the need for restricted attention spans - meaning not only having to manually select the order in which to generate pixels but also having to manually select which pixels to attend to in order to generate those pixels [50].

4.3.3 Masked Language Models

Bidirectional Language Models To overcome the limited context inherent in autoregressive models, a new type of language model known as bidirectional language models (biLMs) emerged. biLMs are the foundation of successful models such as ELMo [51]. Bidirectional language models perform two passes on each sequence of input tokens. First is a forward pass that attempts to predict tokens in the same way as autoregressive models:

$$p(x_1, ..., x_n) = \prod_{i=1}^{n} p(x_i | x_1, ..., x_{i-1}). \tag{12}$$

This is followed by a backward pass, where tokens are predicted in the reverse direction, giving full context:

$$p(x_1, ..., x_n) = \prod_{i=1}^{n} p(x_i | x_{i+1}, ..., x_n). \tag{13}$$

These predictions are modelled by a type of recurrent neural network known as Long Short-Term Memory networks (LSTMs) [26] that are capable of memorising long sequences in order to utilise long-range contexts when modelling text.

Generalized Language Models After the original introduction and huge success of attention-based transformers for language modelling, there soon followed a wave of new approaches implementing these models with continually better and better results. One such model was OpenAI's GPT (Generative Pre-Training Transformer) [52] as well as its larger successors GPT-2 and GPT-3 [53, 7] . GPT introduced a new methodology for language modelling - firstly a powerful transformer is 'pre-trained' in an unsupervised manner in order to give it a general understanding of the target language(s). After that, the general model has a final linear layer added which is fine-tuned for a variety of downstream tasks such as classification, entailment and similarity. This approach yielded state-of-the-art results on a variety of tasks and is now a popular method for training a variety of language models.

In terms of architecture, GPT discards the encoder from traditional encoder-decoder transformers and only uses the decoder, formed using a series of transformer blocks. During the

unsupervised pre-training stage, sequences that are sampled from the training data are fed as input into the decoder, which then utilises the multi-head attention blocks to attend to all parts of the sequence. The final output is passed into a softmax layer in order to produce a probability distribution over the target tokens, which is optimised autoregressively using the negative log-likelihood:

$$L(\boldsymbol{x}) = \sum_i \log P(x_i | x_{i-k}, ..., x_{i-1}; \theta), \tag{14}$$

where k is the size of the context-window, i.e. how far the transformer can "see" in order to attend to surrounding tokens, and θ is the model parameters.

BERT - The original Masked Language Model An improvement upon and direct descendent of GPT, BERT (Bidirectional Encoder Representations from Transformers) takes the pre-training and fine-tuning training processes of GPT and applies them in a bidirectional manner similar to ELMo. However, instead of using two separate passes like ELMo, BERT introduces the concept of **Masked Language Models** (MLMs) [12].

During the pre-training stage of training an MLM such as BERT, a transformer-based architecture is trained in an unsupervised manner on free-form text sampled from large datasets. Each input sequence then has 15% of it's tokens randomly replaced with a [MASK] token, and the aim of the model is to correctly predict the original tokens. Training in this manner allows the MLMs to have full, bidirectional context to make predictions without the need for two separate passes. This additional context unlocks the full representation-learning potential in the underlying transformers and allows for incredibly impressive results when the general model is fine-tuned for downstream tasks.

The other primary difference between BERT and GPT is the fact that BERT's architecture is just a transformer encoder, unlike GPT which uses a decoder. The reason for this is that the original decoder, as introduced in [74], utilises masking for autoregressive decoding, whereas BERT is non-autoregressive and as such does not need to use the masked transformer block, which when removed from the decoder leaves an encoder architecture.

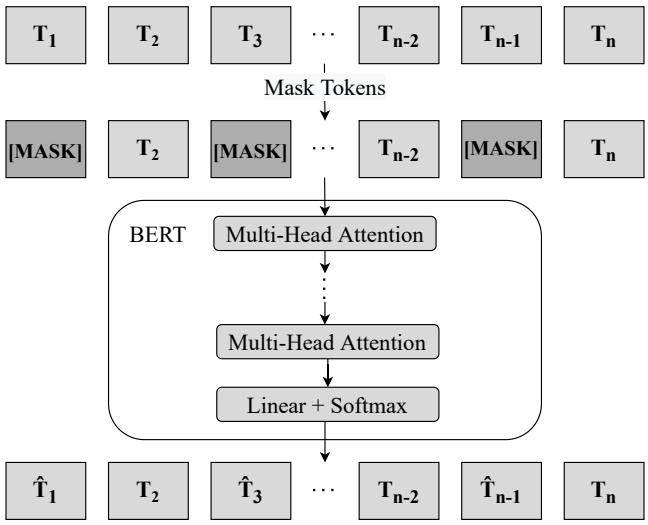


Figure 5: The BERT training process. First an input sequence has some of its tokens randomly masked, while another smaller percentage of tokens are replcaed with other random tokens. BERT takes these masked & shuffled tokens as input and predicts the original values of the [MASK] tokens using bidirectional, multi-head, scaled dot-product attention.

4.4 Denoising Diffusion

The final group of generative model architectures necessary to introduce for the purpose of this thesis are denoising diffusion probabilistic models (DDPMs) - a family of flexible, highly expressive and stable generative models [24, 48, 64]. In terms of sample quality, these models have come to rival other state of the art generative methods such as GANs and flow models, without the instability of GANs nor the architectural restrictions of flow models [19, 57]

DDPMs use a very unique approach to generative modelling, based upon concepts in non-equilibrium thermodynamics [65]. They are technically a type of latent-variable model, and function by gradually transitioning between the input data space and a latent space of the same dimensionality, before then reversing that process. To achieve this, a Markov chain is defined over a large number of steps, T, with the start step $t = 0$ taking a data sample $\boldsymbol{x}_0$ as input and the final step $t = T$ producing a latent-space sample $\boldsymbol{x}_T$. At each step, a small amount of Gaussian noise is added to the input to produce $\boldsymbol{x}_t$, a slightly noisier version of $\boldsymbol{x}_{t-1}$. As $T \to \infty$, the latent space becomes equivalent to a Gaussian distribution and becomes possible to sample from, similar to the latent space of VAEs (except using the same dimensionality as the data). DDPMs are optimised to reverse this diffusion process and gradually denoise sampled latents to produce data samples. This makes it possible to sample from the latent space and generate data samples without the need for assumed priors or adversarial training, resulting in expressive and high-quality samples when the models are trained appropriately.

Forward Diffusion Process The forward diffusion process, $q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)$, where T is the number of time steps and $\boldsymbol{x}_0 \sim q(\boldsymbol{x})$ is the data sampled from the training dataset, is defined using a fixed variance schedule $\beta_1, ..., B_T$:

$$q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) = \prod_{t=1}^{T} q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}), \quad q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t \boldsymbol{I}). \tag{15}$$

This diffusion process has a number of useful properties, the most important of which is that it permits sampling $\boldsymbol{x}_t$ at an arbitrary time step t in closed form. Using $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, (1-\bar{\alpha}_t)\boldsymbol{I}). \tag{16}$$

Reverse Process Given a DDPM with a fixed forward diffusion process q, and a learned reverse process p_θ, it is possible to optimise the parameters θ by optimising the variational bound on the negative log-likelihood:

$$\mathbb{E}[-\log p_\theta(\boldsymbol{x}_0)] \leq \mathbb{E}_q \left[-\log \frac{p_\theta(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \right] = \mathbb{E}_q \left[-\log p(\boldsymbol{x}_T) - \sum_{t \geq 1} \log \frac{p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)}{q(\boldsymbol{x}|\boldsymbol{x}_{t-1})} \right] = L. \tag{17}$$

As it possible to sample $q(\boldsymbol{x}_t|\boldsymbol{x}_0)$ (Eq. (16)) at an arbitrary time step t, efficient training can be performed by optimising random selected timesteps. Training can be further stabilised through reduced variance by rewriting Eq. (17) as:

$$L = \mathbb{E}_q \left[\underbrace{D_{KL}(q(\boldsymbol{x}_T|\boldsymbol{x}_0)||p(\boldsymbol{x}_T))}_{L_T} + \sum_{t>1} \underbrace{D_{KL}(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)||p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t))}_{L_{t-1}} - \underbrace{\log p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1)}_{L_0} \right]. \tag{18}$$

By rewriting the loss in this way, it becomes possible to directly compare the reverse process $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ against the tractable forward posteriors using KL divergence.

There are obvious similarities between the VAE variational lower bound defined in Eq. (1) and Eq. (18). Specifically, attempting to minimise the KL divergence between a posterior and normally distributed prior for L_t. This makes sense as VAEs also attempt to encode data into a Gaussian latent space, just in a simpler (and less powerful way) than DDPMs.

There are a number of different methods used to train a DDPM using the loss function given in Eq. (18). Details are omitted here for the sake of simplicity, but for image generation, one of the most popular of such methods is train a convolutional neural network (such as a U-net [58]) to denoise any randomly given $\boldsymbol{x}_t \sim q(\boldsymbol{x}_t|\boldsymbol{x}_0)$ and produce an estimate of $\boldsymbol{x}_0$, $\hat{\boldsymbol{x}}_0$. This estimate $\hat{\boldsymbol{x}}_0$ can then be used to compute an estimate $\boldsymbol{\epsilon}_\theta$ of the noise added at timestep t, and that noise can be removed from $\boldsymbol{x}_t$ to give $\boldsymbol{x}_{t-1}$. During sampling time this is repeated iteratively until a data sampled $\boldsymbol{x}_0$ is produced.

While DDPMs are globally-consistent, simple to train and highly expressive generative models, they do come with their downsides. Most prominently is the fact that sampling DDPMs can be a very slow process, as it requires T passes through the neural network per batch of samples. However, many intuitive methods have been developed that allow pre-trained DDPMs to be sampled much more efficiently using significantly fewer steps, at the cost of a slight reduction in sample quality [64, 75]. The final downside of the approaches outlined so far is that they are only applicable in continuous data spaces due to the use of a continuous Gaussian distribution for the forward diffusion process, something that previous work and this thesis aims to address.

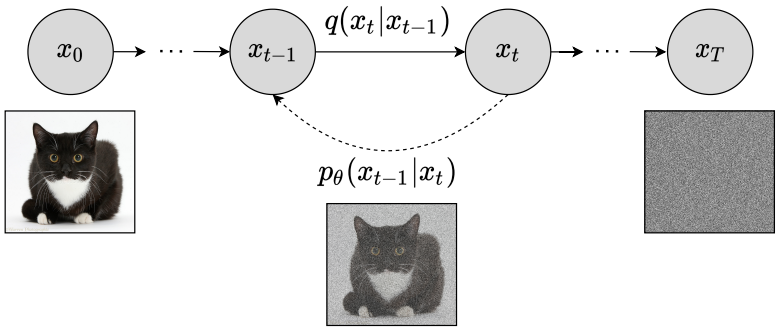


Figure 6: A DDPM markov chain. The fixed forward process q adds a small amount of random noise at each step and eventually converges to a Gaussian distribution at step T. The reverse process p_θ learns to gradually remove this noise to produce samples from a Gaussian latent space.

4.4.1 Discrete Diffusion

While DDPMs are not designed to function on discrete data, they can be adapted to do so. One particularly effective approach for this is known as *Multinomial Diffusion* [27]. Multinomial diffusion utilises the fundamentals of DDPMs and applies them on categorical data to give a likelihood-based generative model for discrete data.

Forward Process For Multinomial Diffusion it is necessary to encode $\boldsymbol{x}_t$ in a one-hot format $\boldsymbol{x}_t \in \{0,1\}^{(K\times D)}$, where K is the number of categories and D is the data dimension. For the purposes of explanation $\boldsymbol{x}_t$ will be described as an image (although diffusion can be used on other types of data) and each category k as a specific colour value that the pixel can take, with each pixel $\boldsymbol{x}_{ti}$ corresponding to a single one-hot vector of category k, with $x_{tik} = 1$ and $x_{tij} = 0$ for $j \neq k$.

Encoding the data in this way makes it possible to define a forward diffusion process using one-hot categorical distributions, where at each time step t the distribution has a β_t chance of uniformly re-sampling each pixel:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{C}(\boldsymbol{x}_t|1-\beta_t)\boldsymbol{x}_{t-1} + \beta/K). \tag{19}$$

Unlike the Gaussian latent space of DDPMs, as $T \to \infty$ in multinomial distributions the latent space converges to a uniform categorical distribution, once again allowing for easy sampling of $\boldsymbol{x}_T$ to be used as a starting point of the denoising sampling process.

As with DDPMs in Eq. (16), the above equation Eq. (19) forms a Markov chain and can be used to sample $q(\boldsymbol{x}_t|\boldsymbol{x}_0)$ at any arbitrary time step t:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{C}(\boldsymbol{x}_t|\bar{\alpha}_t\boldsymbol{x}_0 + (1-\bar{\alpha}_t)/K), \tag{20}$$

where $\bar{\alpha}_t$ is defined in the same way as in DDPMs.

Reverse Process Once again omitting technical details for the sake of simplicity, the reverse process in multinomial diffusion utilises the same process as that used in DDPMs. During training a neural network learns to estimate $\hat{\boldsymbol{x}}_0$ given any input $(\boldsymbol{x}_t, t)$. Eq. (19) and Eq. (20) can then use this value to compute an estimate of the 'noise' added at the time step t. The 'noise' in this case is not Gaussian in nature but rather an alteration to the underlying categorical distribution probabilities, and it is these changes that are estimated and removed before re-sampling $\boldsymbol{x}_{t-1}$

While this approach to applying diffusion to discrete data spaces does work to a degree, the sample results produced in the original work are not particularly impressive, especially when compared to the results produced by DDPMs and other continuous diffusion approaches. The primary reason for this is that the uniform re-sampling is not equivalent to adding continuous Gaussian noise. By entirely re-sampling a pixel, all the information in that pixel is completely destroyed and the model can no longer use it to learn anything useful, however the model does not know this as the pixel still has a value corresponding to a valid colour. This means at later time steps the model will be unable to differentiate between the original pixels and already re-sampled pixels, making it much harder to extract the necessary information to learn an effective reverse process. Sec. 5.3 introduces a new discrete diffusion process that aims to solve this problem.

Figure 7: Images generated by MLM diffusion models trained on 256x256 datasets: LSUN Churches, FFHQ, and LSUN Bedroom.

4.5 High-Resolution Image Generation

The field of high-resolution image generation research has developed rapidly over the past decade. Starting in 2014, GANs were the dominant choice of architecture [19, 35, 34]. GANs demonstrate impressive performance on image-generation for low resolutions but are unstable to train and scale quadratically in computational cost with respect to image resolution. VAEs were also popular [37] at this time. However, they struggle to sufficiently capture the complexity of high-resolution image datasets due to the loss of information when applying a Gaussian prior.

Since 2020, there has been much interest in the use of transformers [74, 14] and diffusion models [24, 25, 66] for various high-resolution image tasks. Transformers' attention mechanisms are very powerful but scale very poorly to longer sequences and so cannot be easily utilised for high-resolution generation. Diffusion models, while demonstrating incredibly impressive results and stable training, only operated successfully in the pixel-level space and thus were also constrained significantly in the resolutions they could operate on.

This thesis and the paper published alongside it [3] are based upon recent work in developing generative models on discrete latent spaces [17, 73]. As discussed in the Sec. 4.1.2, these models first learn a discrete latent representation to reduce the spatial dimension before then training a generative model on this latent space. Due to this reduction in the spatial dimension, the sequence length is reduced and the power of transformers can be applied to the task of image generation. This work was written adjacently to a number of similar works investigating various methods of approaching this technique [16, 8].

Method

5.1 Overview

The method proposed in this section investigates a novel two-stage generative modelling approach (Sec. 4.1.4), with the final goal of generating high-resolution, high-fidelity and globally-consistent images.

The first training stage, where compressed, discrete representations of high-resolution images are learned, is implemented using the VQGAN architecture (Sec. 4.1.5). The effectiveness of these models is investigated thoroughly, and this thesis proposes changes to the training approach of VQGANs in order to improve their training stability and representation capabilities.

For the second training stage, where a sampleable prior over the discrete latent space is learned, this thesis introduces a novel approach that uses Masked Languages Models (Sec. 4.3.3) for the underlying architecture of a Discrete Diffusion process (Sec. 4.4.1). Critically, this sampling process is non-autoregressive in nature, ensuring that all generated latents have global access to information in the rest of the sample, improving image-consistency and therefore sample quality.

5.2 Learning Effective Discrete Representations

5.2.1 VQGAN Training

The VQGAN architecture, as introduced by Esser et al. [17], is able to produced compressed, discretised representations of images through a combination of a convolutional autoencoder architecture and a patch-based discriminator, as discussed in Sec. 4.1.5. For compressed image representations with a spatial dimension of 8×8 or less, it is also possible to add self-attention blocks at these layers in order to improve the representation quality of the learned latents. It is technically possible to add these layers to larger representations but training is significantly slowed and not worth the marginal improvements in reconstruction quality achieved.

Datasets Used For the experiments carried out in this thesis, the VQGAN is trained on 3 different datasets: FFHQ, LSUN-Churches and LSUN-Bedrooms.

FFHQ (Flickr-Faces-HQ) and LSUN Churches (Large-Scale Understanding) are high-resolution image datasets commonly used for benchmarking deep generative models in high-resolution image generation. Each LSUN category contains 50,000 training images and FFHQ has 70,000, with both of them being loaded in at a resolution of 256x256 pixels for all experiments, although FFHQ can be used at resolutions up to 1024x1024.

Compression During the first training stage, the VQGAN is implemented to downsample the datasets to a discrete latent space of pre-determined dimensionality. The 256x256 images from the high-resolution datasets are compressed to 16x16 latent spaces, reducing them to 0.391% of their original spatial size, with each latent embedded to a dimension of 256. Despite this massive compression, the VQGAN is still able to reconstruct these images with minimal loss of image quality, as shown in Sec. 6.6.

Training Objective The training objective is firstly based upon that of VQVAEs, given in Eq. (4) and repeated here for readability:

$$\mathcal{L}_{VQ}(\theta, \phi, \boldsymbol{x}) = \log p_\theta(\boldsymbol{x}|q(\boldsymbol{z})) + ||\text{sg}[\boldsymbol{z}] - q(\boldsymbol{z})||_2^2 + \beta||\boldsymbol{z} - \text{sg}[q(\boldsymbol{z})]||_2^2. \tag{21}$$

Here, $\log p_\theta(\boldsymbol{x}|q(\boldsymbol{z})) = \mathcal{L}_{\text{rec}}$ represents the reconstruction loss. During training this is computed as an L1 loss. Therefore, by defining $\hat{\boldsymbol{x}}$ as the reconstructed output of the VQGAN, $\mathcal{L}_{VQ}$ can be rewritten as:

$$\mathcal{L}_{VQ}(\theta, \phi, \boldsymbol{x}) = ||\boldsymbol{x} - \hat{\boldsymbol{x}}||_1^1 + ||\text{sg}[\boldsymbol{z}] - q(\boldsymbol{z})||_2^2 + \beta||\boldsymbol{z} - \text{sg}[q(\boldsymbol{z})]||_2^2. \tag{22}$$

A VQGAN also contains a patch-based discriminator, D. This means that an additional, adversarial loss term $\mathcal{L}_{GAN}$ is required to train the discriminator:

$$\mathcal{L}_{GAN} = \log D(\boldsymbol{x}) + \log(1 - D(\hat{\boldsymbol{x}})). \tag{23}$$

Taking all of this together, the final loss function can be computed as a linear combination of Eq. (22) and Eq. (23):

$$\mathcal{L} = \min_{\theta,\phi,\mathcal{E}} \max_{D} \mathbb{E}_{\boldsymbol{x}\sim p_d} \left[\mathcal{L}_{VQ} + \lambda \mathcal{L}_{GAN}\right], \tag{24}$$

where λ is an adaptive weighting for the discriminator, used to help balance the training of the autoencoder and the discriminator and minimise mode collapse:

$$\lambda = \frac{\nabla_{G_L}[\mathcal{L}_{\text{rec}}]}{\nabla_{G_L}[\mathcal{L}_{\text{GAN}}] + \delta}. \tag{25}$$

5.2.2 Improving VQGAN

During the initial experiments it was noticed that the VQGAN suffered from some instability issues during training, especially on datasets using smaller images. Further investigation revealed that the cause of this instability was the adaptive weighting λ taking on very high values early on during training. This causes excessive mode collapse and codebook collapse, resulting in poor quality reconstructions early on during training that take a long time to recover from. To remedy this, a simple technique of limiting λ to some maximum value λ_{max} resulted in massively improved stability and faster convergence.

Another improvement used for the experiments in Sec. 6 is the use of differentiable augmentations (DiffAug) [78]. It has been shown that data augmentation is very effective at improving the quality of sample in generative methods [32]. By applying a set of differentiable data augmentations, T, to all discriminator inputs, it is possible to further improve the quality of the discriminator and thus the reconstructions.

By taking both of the above improvements into consideration, both a new $\mathcal{L}_{GAN}$ objective and a new definition for the adaptive weighting λ are defined that yield improvements when training on both low and high-resolution datasets:

$$\mathcal{L}_{\text{GAN}} = \log D(T(\boldsymbol{x})) + \log(1 - D(T(\hat{\boldsymbol{x}}))), \tag{26}$$

$$\lambda = \min\left(\frac{\nabla_{G_L}[\mathcal{L}_{\text{rec}}]}{\nabla_{G_L}[\mathcal{L}_{\text{GAN}}] + \delta}, \lambda_{\text{max}}\right). \tag{27}$$

5.3 Discrete Diffusion using BERT

This section introduces a new discrete diffusion method, MLM Diffusion, that aims to build and improve upon DDPMs and their discrete counterparts. As the name implies, unlike the convolutional U-Nets or similar architectures used by other diffusion models, the underlying architecture in this proposed method is a bidirectional transformer. By training this transformer using approaches inspired by those used to train Masked Language Models, it becomes possible to define a reversible diffusion process that takes discrete data as input and diffuses gradually

to a single, fully-masked point. When paired with a VQGAN in a two-stage training process, this diffusion method is effective for sampling the discrete latent space to efficiently generate high-quality, globally-consistent high-resolution images.

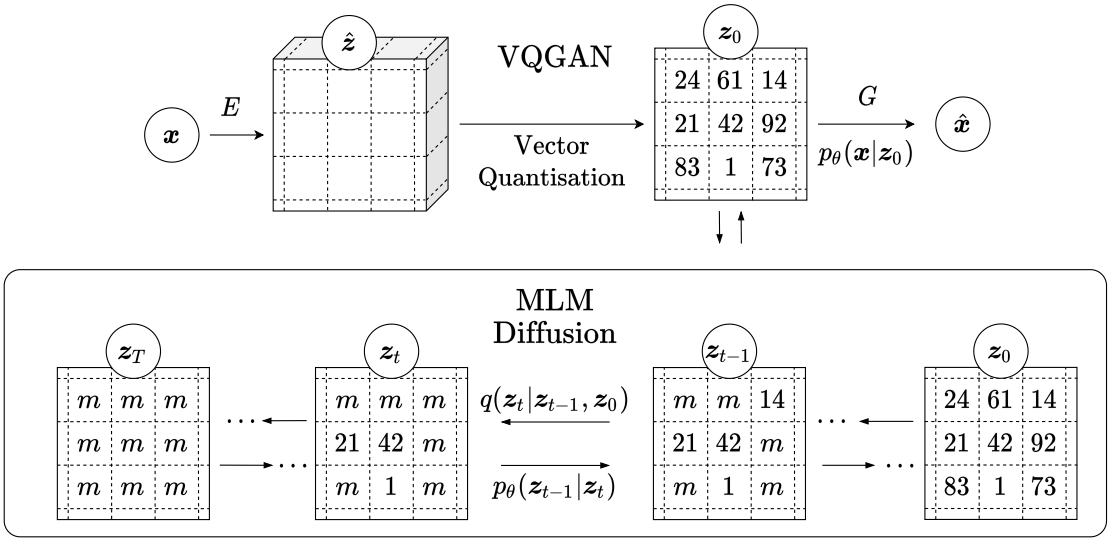


Figure 8: MLM Diffusion Architecture - Discrete tokens generated by a trained VQGAN and gradually replaced with [MASK] tokens ("m" in the diagram) until eventually the entire latent is fully masked.

5.3.1 Forward Diffusion

In order to define a forward diffusion process that encourages training in a similar manner to that used by MLMs, this thesis introduces the concept of a 'masking schedule'. Each type of masking schedule, q, is defined over a number of timesteps, T, which can be any number equal to or larger than the dimensionality of the latent space, D.

For both schedules the same conditions are assumed: a diffusion process over T time steps and an initial latent sample $\boldsymbol{z}_0 = \{z_0, z_1, ..., z_N\}$ from the VQGAN encoder, where N is the dimension of the latent space. Each latent z_i is an integer value that corresponds to an index from the VQGAN discrete codebook.

Probabilistic Masking Schedule For the probabilistic masking schedule, at each time step t each unmasked latent z_i is masked with a probability of $\frac{t}{T}$. This means that initially most latents are unmasked, however as $t \to T$ the probability of any given token being masked approaches and eventually reaches 1. Therefore, the forward diffusion process $q(\boldsymbol{z}_t|\boldsymbol{z}_0)$ is defined as follows:

$$\begin{aligned} q([\boldsymbol{z}_t]_i = m|\boldsymbol{z}_0) &= \frac{t}{T}, \\ q([\boldsymbol{z}_t]_i = [\boldsymbol{z}_0]_i|\boldsymbol{z}_0) &= 1 - \frac{t}{T}. \end{aligned} \tag{28}$$

The non-deterministic nature of Denoising Diffusion is what enables it to so effectively model a data distribution and so mustn't be disregarded. Critically, the probabilistic masking schedule introduces a stochastic element to the training process, similar to Gaussian noise in continuous diffusion, that the transformer must account for when attempting to denoise. Furthermore, it naturally pairs with certain training objectives to improve the quality of training and samples, as will be shown in Sec. 6.

Fixed Masking Schedule This masking schedule is still a gradual process, with step T being fully-masked and step 0 being fully unmasked. However, in contrast to the probabilistic masking schedule, each step is not fully probabilistic, instead masking precisely $\lfloor \frac{t}{T} \times D \rfloor$ tokens at each timestep t, where D is the size of the latent space, giving the following definition:

$$\begin{aligned} q([\boldsymbol{z}_t]_i = m|\boldsymbol{z}_0) &= \frac{\lfloor \frac{t}{T} \times D \rfloor}{D}, \\ q([\boldsymbol{z}_t]_i = [\boldsymbol{z}_0]_i|\boldsymbol{z}_0) &= 1 - \frac{\lfloor \frac{t}{T} \times D \rfloor}{D}. \end{aligned} \tag{29}$$

Using the fixed masking schedule has a number of effects on the training process. Most importantly, the variance in training is reduced. Using the fixed masking schedule, the number of masked tokens at time step t will always be the same, making training smoother, especially when normalising the loss with respect to the number of masks. This reduction in randomness does not make the process deterministic or autoregressive: the tokens to be masked at each step are still selected at random, it is only the number of tokens that are masked at each step that is fixed.

5.3.2 Reverse Process

MLM Diffusion uses the same approach as DDPMs: a neural network is trained to reverse some fixed forward diffusion process in order to produce samples from a data distribution (in the case of this thesis, the discrete, compressed space produced by a VQGAN). MLM Diffusion utilises the power of bidirectional transformers for this reverse process. By training the transformer in a BERT-like manner, it becomes possible to produce samples from the learned distribution by unmasking from a fully-masked starting point.

Training The the training procedure for these models follows the following steps:

1. Sample a new latent $\boldsymbol{z}_0 \sim \mathcal{Z}$, where $\mathcal{Z}$ is the latent distribution produced by the VQGAN in the first training stage.
2. Select a random time step t, $0 < t \leq T$.
3. Randomly replace tokens in the sampled latent $\boldsymbol{z}_0$ with [MASK] tokens according to one of the masking schedules from Sec. 5.3.1.
4. Pass the masked latent into the bidirectional transformer and use it to predict the original values of the [MASK] tokens. This produces an estimate of the sampled latent, $\hat{\boldsymbol{z}}_0$.
5. Calculate the loss between $\hat{\boldsymbol{z}}_0$ and $\boldsymbol{z}_0$ and use this to perform an optimisation step on the transformer's parameters.
6. Repeat steps 1 - 5 until loss has converged

This process ensures that the model is trained evenly across all time steps and thus is capable of producing useful estimates even when the majority of the latents are masked. This is in contrast to BERT, which only ever masks 15% of tokens in its training sequences.

Sampling Sampling from MLM Diffusion models is a very similar process to that used in DDPMs. The following relationship between the forward and reverse processes allows the model to make a single reverse step:

$$p_\theta(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t) \propto q(\boldsymbol{z}_{t-1}|\hat{\boldsymbol{z}}_0) p_\theta(\hat{\boldsymbol{z}}_0|\boldsymbol{z}_t), \tag{30}$$

where θ represents the parameters of the transformer and q is the forward diffusion process.

This relationship makes it relatively simple to sample from the MLM diffusion models using the following process:

1. Create a fully-masked latent of appropriate size, $\boldsymbol{z}_T$
2. Pass the masked latent through the trained unmasking transformer to produce $\hat{\boldsymbol{z}}_0$.
3. Generate $\boldsymbol{z}_{t-1}$ by re-masking $\hat{\boldsymbol{z}}_0$ using the chosen masking schedule q.
4. Repeat steps 2 - 3 until a sample z_0 is generated.

Algorithm 1: Diffusion MLM Training

Initial args: T, $\mathcal{Z}$;
for *step in training_steps* **do**
  $\boldsymbol{z}_0 \sim \mathcal{Z}$;
  $t \sim \text{Uniform}(T)$
  $\boldsymbol{z}_t \sim q(\boldsymbol{z}_t|\boldsymbol{z}_0)$;
  $\hat{\boldsymbol{z}}_0 \sim p_\theta(\boldsymbol{z}_0|\boldsymbol{z}_t)$
  $\mathcal{L} \longleftarrow ||\hat{\boldsymbol{z}}_0 - \boldsymbol{z}_0||$
  Take gradient descent step on $\nabla_\theta \mathcal{L}$
end

Algorithm 2: Diffusion MLM Sampling

Initial args: q, p_θ;
$\boldsymbol{z}_0 \longleftarrow$ ([MASK], ..., [MASK]) $\in \mathbb{R}^D$;
$t \longleftarrow T$
while $t > 0$ **do**
  $\boldsymbol{z}_{t-1} \sim q(\boldsymbol{z}_{t-1}|\boldsymbol{z}_0)$
  $\hat{\boldsymbol{z}}_0 \sim p_\theta(\boldsymbol{z}_0|\boldsymbol{z}_{t-1})$
  $\boldsymbol{z}_0 \longleftarrow \hat{\boldsymbol{z}}_0$
  $t = t - 1$
end
Return z_0

Figure 9: Pseudo code for training (left) and sampling (right) MLM Diffusion models

Benefits The MLM Diffusion training and sampling processes have a number of key benefits over other methods used for learning a prior over a discrete latent space. Firstly, unlike autoregressive methods, it is globally-coherent: no predefined ordering over the latents is required at any training or sampling step, and all unmasked latents have access to the information stored by all other unmasked latents. Therefore, these models are able to generate samples that are consistent across the entire latent space, not just in local clusters. Furthermore, the masking process means that no unnecessary information is passed to the transformer, whereas in multinomial diffusion 're-sampled' latents would influence and slow the transformer's training. Finally the use of a diffusion process makes it trivial to speed up sampling by reducing the number of sampling steps, i.e. by unmasking more latents per time step.

5.3.3 Training Objectives

An important decision when training diffusion models is selecting a training objective. As each diffusion step, $\mathcal{L}_t$, can be optimised individually, it is possible to selectively optimise some steps more or less than others. This selective optimisation is achieved through giving a higher weighting to more important steps, and a lower weighting to the less important ones. This thesis investigates 3 such weightings: ELBO, Mask-Normalised Loss, and Re-weighted ELBO.

ELBO The first and most straightforward objective is to train the Diffusion MLMs using the same objective as that used for some DDPMs [63]: maximising the ELBO through optimising the variational bound on the negative log-likelihood. This is also defined in Eq. (18). When implemented, the cross-entropy between the original values of $\boldsymbol{z}_0$ and the predicted values of the

[MASK] tokens produced by the bidirectional transformer is calculated before being normalised with respect to the time step t, giving an equal weighting to all steps $\mathcal{L}_t$

Mask-Normalised Loss Instead of normalising with respect to the time step t, the MLM Diffusion method enables the possibility of normalising with respect to precisely how many tokens were masked. Normalising each $\mathcal{L}_t$ term in this way significantly reduces the variance in the training objective when compared to optimising the ELBO directly. This training objective is called Mask-Normalised Loss and can be defined as:

$$\mathcal{L}_{\text{MN}} = -\,\mathbb{E}_{q(\boldsymbol{z}_0)}\left[\mathbb{E}_{t\in[1,\dots,T]}\left[\frac{1}{k}\,\mathbb{E}_{\boldsymbol{x}_t}\left[\sum_{i,\ [\boldsymbol{x}_t]=[\text{MASK}]}\log p_\theta([\boldsymbol{x}_0]_i|\boldsymbol{x}_t)\right]\right]\right], \tag{31}$$

where k is the number of masks, which is determined using the fixed masking schedule as defined in Eq. (29).

This training objective is implemented in the same way as optimising the ELBO, however each term is simply normalised by the sum of the number of [MASK] tokens instead of the time step.

Re-weighted ELBO The two previously introduced objectives normalise with respect to how much of the latent has been masked, either through the use of the number of time steps, t, or the number of [MASK] tokens, k. This results in each loss step $\mathcal{L}_t$ contributing approximately equally to the final loss sum at each training step. However, it can be argued that these training objectives, while effective, leave much room for improvement and do not enable the transformer to optimise as effectively as possible. Instead, in order to train more efficiently and with the actual goal of the transformer in mind, a new objective can be introduced that re-weights the $\mathcal{L}_t$ terms.

As the time step, t, progress from 0 to T, more and more of the latent $\boldsymbol{z}_t$ becomes masked. The more of the latent that is replaced with [MASK] tokens, the less information the transformer has access to and thus the more difficult a task it becomes to accurately estimate $\hat{\boldsymbol{z}}_0$. Attempting to predict $\boldsymbol{z}_0$ when the latent is fully-masked or close to fully-masked is effectively impossible, and the results will be near random but affect the long-term outcome of the sample. Meanwhile, when t is close to 0, predicting the masked tokens will be an easy, near trivial task that only affects the finer details of the sampled latent. The steps in between, and specifically the steps close to $t = \frac{T}{2}$, are a mix of these two extremes, where the transformer has access to significant amounts of information and the predictions that are made affect the global structure of the samples.

Following this logic, the final proposed training objective weights each $\mathcal{L}_t$ such that the steps closest to $t = \frac{T}{2}$ contribute significantly more to the final loss, whereas the trivial tasks near $t = 0$ and impossible tasks near $t = T$ receive little-to-no weighting. To do this, each $\mathcal{L}_t$ is multiplied by $(1 - \frac{t}{T})$ but no normalisation is applied. This coefficient allows for the weight to linearly increase as t gets closer to 0, but the lack of normalisation means that as fewer tokens are masked, the contribution to the loss sum decreases and thus affects the optimisation less. Named Re-weighted ELBO, this training objective strikes a balance that allows for more efficient and effective training, and is formally defined as:

$$\mathcal{L}_{\text{MN}} = -\,\mathbb{E}_{q(\boldsymbol{z}_0)}\left[\mathbb{E}_{t\in[1,\dots,T]}\left[(1-\frac{t}{T})\,\mathbb{E}_{\boldsymbol{x}_t}\left[\sum_{i,\ [\boldsymbol{x}_t]=[\text{MASK}]}\log p_\theta([\boldsymbol{x}_0]_i|\boldsymbol{x}_t)\right]\right]\right]. \tag{32}$$

5.3.4 Generating Super-Resolution Images

An important property of convolutional neural networks is that they preserve spatial information when compressing image data to smaller dimensions. This results in the latents generated by the VQGAN being highly spatially correlated with the output image pixels in the same relative spatial locations. Using this information, it becomes possible to use the MLM diffusion method to generate arbitrarily-sized images by generating latents of an appropriately-chosen size and shape. This thesis proposes an approach that allows high-fidelity images that are significantly larger than those used in the training dataset to be generated.

Firstly, to generate a super-resolution image, a large $a \times b$ array of mask tokens, $\bar{\boldsymbol{z}}_T = m^{a \times b}$, is initialised that corresponds to the desired size of image one wishes to generate. For example, as the 256×256 pixel training images are compressed to a latent size of 16×16, generating images of resolution 512 x 256 requires latents of size 32×16. Or for generating images with a resolution of 1024×384, latents of size 64×24 would be used.

In order to capture the maximum context when approximating $\bar{\boldsymbol{z}}_0$, the denoising network is applied to all subsets of $\bar{\boldsymbol{z}}_t$ with the same spatial size as the network's usual inputs, aggregating estimates for each location. Specifically, using $c_j(\bar{\boldsymbol{z}}_t)$ to represent local subsets, the denoising distribution is approximated as a mixture:

$$p([\bar{\boldsymbol{z}}_0]_i | \bar{\boldsymbol{z}}_t) \approx \frac{1}{Z} \sum_j p([\bar{\boldsymbol{z}}_0]_i | c_j(\bar{\boldsymbol{z}}_t)), \tag{33}$$

where Z is the number of subsets, and the sum is over subsets c_j that contain the i^{th} latent.

For extremely large images, this solution can require a very large number of function evaluations. However, the sum can be approximated by striding over latents with a step > 1 or by randomly selecting positions. Samples generated using this approach can be seen in Fig. 14 and Fig. 1.

5.3.5 Implementation Details

Architecture The architecture of the transformer trained for all experiments is implemented using an adapted version of minGPT [33]. This model is a PyTorch implementation of the techniques used by the popular and powerful GPT-3 architecture [7], but using significantly fewer parameters. Using this model, it becomes possible to train a MLM Diffusion sampler on 16×16 latents on a single NVIDIA 2080ti.

Training Hyperparameters The hyperparameters selected for the VQGAN and Diffsion Transformer and the tuning process are discussed in the appendix Sec. 9.1.

- Training steps and time - the MLM Diffusion models were all trained for a minimum of 1 million steps, with each step taking approximately 0.2s (giving a 55 hour training time). The FFHQ models converged by step 700,000, whereas LSUN Churches and Bedrooms stil had not converged at this point, but were nonetheless already producing impressive results. The best sampler models (including autoregressive) for each dataset were selected using early-stopping: a method whereby training is stopped when the validation loss is at its lowest, just before overfitting begins to occur.

- Model size - All architectural parameters have an effect on the size of the models. The configuration chosen for the transformer gives a model with 77 million parameters, and the 68 million for the VQGAN. This results in a total of 145 million parameters across both training stages.

Experiments

6.1 Sample Quality

In this section samples are taken from trained MLM diffusion models and evaluated quantitatively against other state-of-the-art approaches through the use of a number of metrics: Fréchet Inception Distance (FID) and Precision, Recall, Density and Coverage (PRDC), with discussion on the benefits and downsides of each. Sample images can be seen in (Fig. 10, Fig. 15 & Fig. 16), which qualitatively show that MLM diffusion models produces high-quality, globally-consistent and diverse images.

Figure 10: Non-cherry picked, $t = 0.9$, 256x256 LSUN Churches samples.

6.1.1 Metrics

Fréchet Inception Distance (FID) Introduced in 2017 [22], FID is the most commonly-used metric for evaluating generative image models at the time of writing. It functions by projecting both real images and fake samples into embedded feature spaces using a pre-trained convolutional neural network such as Inception v3 [67], computing the mean and variance of the embedded distributions (by assuming they are Gaussian) and then computing the Wasserstein-2 distance between the two distributions. The logic for this approach follows that the more similar the generated images are to those from the original data, the more similar the embedded features' distributions will be (i.e. the real images distribution and the generated images distribution) and therefore the lower the calculated FID score will be.

Limitations of FID Metric FID has been found to correlate well with image quality, but it is not without its flaws. The primary issue with FID is its one-dimensional nature. To properly assess the quality of a generative model, one needs to be able to measure both the quality and diversity of the samples produced. While FID does take both of these factors into consideration, using a single-dimensional score means it is impossible to differentiate between how much each factor contributes to the outcome: a low FID score could mean a model produces very high-quality samples, very varied samples, or both. Furthermore, FID unrealistically approximates the data distribution as Gaussian in embedding space and is insensitive to the global structure of the data distribution, resulting in bias towards adversarial generative models [68].

Alternative approaches that address these issues have been developed [5] such as PPL [35], which assesses sample consistency through latent interpolations; IMD [68], which uses all moments to compare data manifolds making it sensitive to global structure; and MTD [1], which compares manifolds in image space. However, none of these have been yet widely-adopted and thus aren't useful for comparative studies such as this thesis.

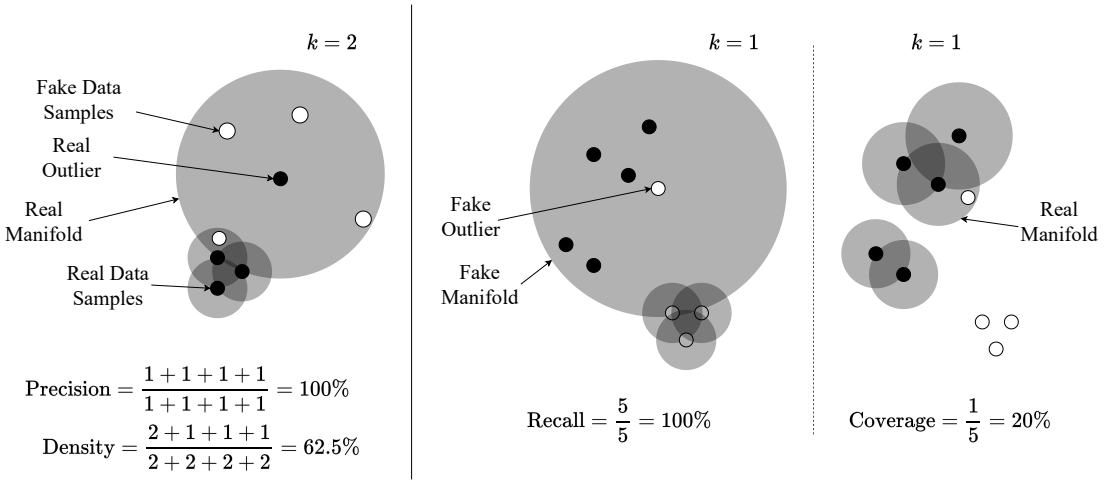


Figure 11: Left diagram: Precision and Density - Density gives significantly poorer scores for sample points that lie near real outliers. Centre and right diagram: Recall and Coverage for the same real and sampled data points: Coverage significantly reduces the influence of fake outliers by calculating its score using the Real Manifold.

Precision, Recall, Density and Coverage (PRDC) Precision and Recall [60] are metrics used to assess generative models that overcome some of the flaws of FID. They are computed by approximating two data manifolds, the real manifold, P_r, and the generated manifold P_g. By comparing samples from these two manifolds it is possible to compute metrics that evaluate the performance of generative models: Precision is measured as the proportion of samples from P_g that lie within P_r, and Recall is the proportion of samples from P_r that lie within P_g.

Critically, the method by which one approximates the data manifolds has a significant effect on the results of the metric computation. An effective and straightforward technique for this is to model the manifold as a set of k nearest-neighbour hyperspheres around samples from the desired distribution [39]. For example, Precision makes a binary decision of whether a fake sample, $x_g \sim P_g$, is contained within the hypersphere of any real image, $x_r \sim P_r$, and takes the proportion of 'yes' decisions to give a precision score bounded between 0 and 1. Similarly to computing FID, this is done in the feature space using a pre-trained image classifier.

A problem with Precision and Recall is that of manifold over-estimation: the phenomena whereby the real or fake manifolds are overestimated by introducing hyperspheres around outlier data that do not accurately represent the true manifold. Density and Coverage [46] are metrics that build upon Precision and Recall and attempt to overcome this problem. Instead of using a single binary decision for each fake sample, Density instead considers the number of real hyperspheres within which each fake sample is contained, and compares that to the nearest-neighbour value, k (i.e. the target number of hyperspheres in which each fake sample should be contained), thus reducing the influence of outlier real samples on the calculated score. Coverage differs from Recall in a similar manner, measuring the proportion of real samples whose neighbourhoods (hyperspheres) contain at least one fake sample.

PRDC has been widely adopted by the deep generative modelling research community for assessing high-resolution image generation as an alternative metric to FID [36, 47, 29, 55]. As such it has been included in the experiments and measurements carried out for the purpose of this thesis. A visualisation of the key differences between Precision, Recall, Density and Coverage can be seen in Fig. 11.

6.1.2 Reduced-Temperature Sampling

To improve sample quality, many generative models are sampled using a reduced temperature or by truncating Normal distributions. This is problematic, as these sampling methods will amplify any biases in the dataset. The impact of temperature on sampling from a model trained on FFHQ is visualised in Fig. 12. For very low temperatures the bias the obvious: samples are mostly front-facing white men with brown hair on solid white/black backgrounds. Exactly how the bias has changed for more subtle temperature changes is less clear, which is problematic. Practitioners should be aware of this effect and it emphasises the importance of dataset balancing.

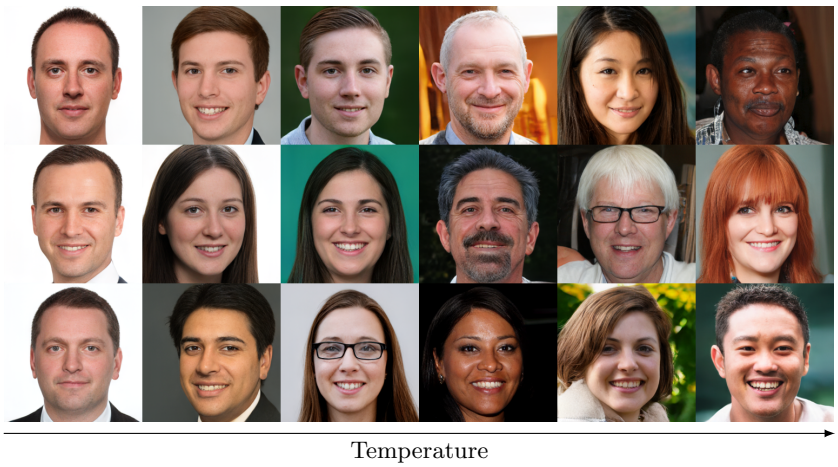


Figure 12: Impact of sampling temperature on diversity. For small temperature changes it is less obvious how bias has changed.

Method	Params	Bed	Church	FFHQ
DDPM [24]	114M	6.36	7.89	-
DCT [47]	448M	6.40	7.56	-
VDVAE [10]	115M	-	-	28.5
TT [17, 16]	600M	6.35	7.81	9.6
ImageBART [16]	2104M	5.51	7.32	9.57
PGGAN [34]	47M	8.34	6.42	-
StyleGAN2 [36]	60M	2.35	3.86	3.8
MLMD ($t = 1.0$)	145M	5.07	5.58	7.12
MLMD ($t = 0.9$)	145M	3.64	4.07	6.11

Table 1: FID for various approaches on FFHQ, LSUN Bedroom, and LSUN Churches. Lower FID signifies higher quality samples.

6.1.3 Results

Samples can be seen in Fig. 7, Fig. 10, Fig. 15 & Fig. 16. These samples qualitatively demonstrate high diversity, fidelity and global consistency.

FID In Tab. 1 the FID is calculated for samples from MLM Diffusion models using the torch-fidelity library [49]. Despite using a fraction of the number of parameters, the proposed method achieves substantially lower FID scores compared to its autoregressive cousins. These scores are competitive with state-of-the-art GAN approaches, while avoiding the known problems associated with GANs, such as mode collapse and highly unstable training.

PRDC In Tab. 2 and Tab. 3 trained MLM Diffusion models are evaluated against other SOTA models in terms of PRDC metrics. Due to limited computing resources, density and coverage scores for DCT [47] and PRDC scores for StyleGAN2 on LSUN Bedroom could not be calculated. On the LSUN datasets MLM Diffusion achieves the highest Precision, Density, and Coverage; indicating that the data and sample manifolds have the most overlap. On FFHQ MLM Diffusion achieves the highest Precision and Recall. In general, when generative models are sampled with lower temperatures to achieve lower FID, this leads to trade-off between precision and recall [36, 55]; since in these experiments FID is calculated with a lower temperature, the effect of reduced-temperature sampling on PRDC was also evaluated. Sampling with temperature leads to improved scores, indicating that MLM Diffusion fits the true data manifold accurately.

Model	P ↑	R ↑	D ↑	C ↑
VDVAE [10]	0.59	0.20	0.80	0.50
TT [17]	0.64	0.29	0.89	0.59
StyleGAN2 [36]	0.69	0.40	1.12	0.80
MLMD ($t = 1.0$)	0.69	0.48	1.06	0.77
MLMD ($t = 0.9$)	**0.73**	**0.48**	**1.20**	**0.80**

Table 2: PRDC results for FFHQ. With temperature sampling MLMD achieves state-of-the-art compared to all approaches with published results on this dataset.

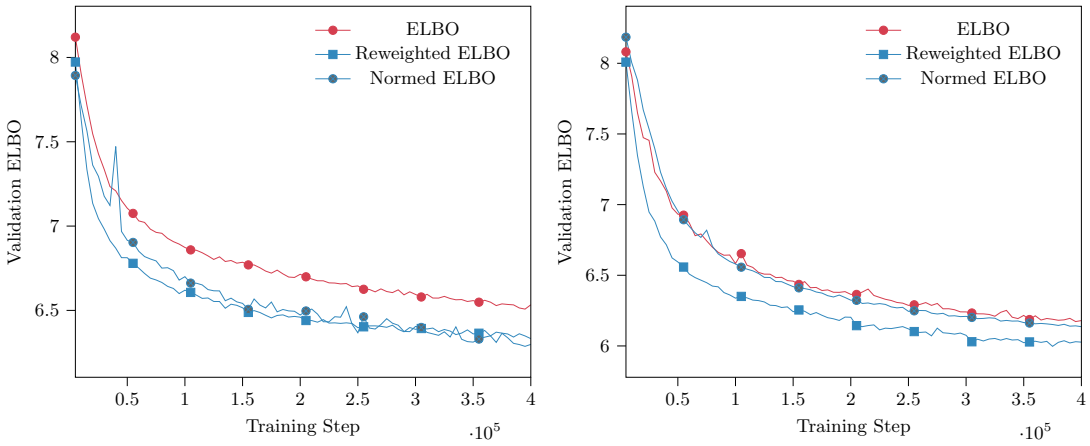
Model	Churches				Bedrooms			
	P ↑	R ↑	D ↑	C ↑	P ↑	R ↑	D ↑	C ↑
DCT [47]	0.60	**0.48**	-	-	0.44	**0.56**	-	-
TT [17]	0.67	0.29	1.08	0.60	0.61	0.33	1.15	0.75
PGGAN [34]	0.61	0.38	0.83	0.63	0.43	0.40	0.70	0.64
StyleGAN [35]	-	-	-	-	0.55	0.48	0.96	0.80
StyleGAN2 [36]	0.60	0.43	0.83	0.68	-	-	-	-
MLMD ($t = 1.0$)	0.70	0.42	**1.12**	0.73	0.64	0.38	1.27	0.81
MLMD ($t = 0.9$)	**0.71**	0.45	1.07	**0.74**	**0.67**	0.38	**1.51**	**0.83**

Table 3: PRDC results for LSUN Churches and Bedrooms (higher is better). With reduced-temperature sampling MLM Diffusion (MLMD) achieves many state-of-the-art results, outperforming all GAN-based methods on Precision, Density and Coverage.

Training Times The MLMD samplers took 10 days to complete training over 1 million steps on a single NVIDIA 2080ti GPU. This is a significant speedup compared to [17], which reports 45.8 days training on an A100 GPU (both faster and with significantly more VRAM).

6.2 Training Objectives

In Sec. 5.3.3 two new training objectives were introduced for discrete diffusion models: Mask-Normalised ELBO and Reweighted ELBO. This section describes the experiments that were carried out to evaluate the efficacy of these training objectives.



(a) Churches (b) FFHQ

Figure 13: Comparison of proposed losses on (a) LSUN Churches and (b) FFHQ. Models trained with reweighted ELBO achieve lower validation ELBO than models trained directly on the ELBO. Interestingly, FFHQ

6.2.1 Validation ELBO

For these experiments, the validation ELBO is computed for each training objective on FFHQ and LSUN Churches. To compute this metric, the dataset is split into two parts - the training dataset and validation dataset, where the validation set is usually much smaller. The models are then trained using only images from the training dataset.

Subsequently, the models' generalisation and quality can be evaluated using the validation dataset. If the models give low ELBO values on the training dataset but high values on the validation dataset (lower is better), then this is a sign of overfitting. This method can be used to compare then generalisation of the models under different training objectives.

6.2.2 Results

Each model was trained for 400,000 steps, using the same setup as detailed in Sec. 5.3.5. The validation ELBO was computed every 5000 steps, iterating over the validation dataset five times to produce an averaged value for the validation ELBO at that time step.

The results from these experiments are visualised in Fig. 13. It is clear that the reweighted ELBO significantly outperforms the standard ELBO on both datasets. By scaling the weights of each time step linearly, decreasing to 0 as $t \to T$, the models are able to better utilise the more information-rich time steps, as explained in Sec. 5.3.3. This demonstrably results in improved validation ELBO on both datasets.

Interestingly, the ELBO normalised with respect to the precise number of [MASK] tokens ("Normed ELBO"), improves validation ELBO on LSUN Churches when compared to the standard ELBO, but has a negligible effect on FFHQ, whereas the reweighted ELBO still gives noticeable improvements. It is difficult to hypothesise the cause of this effect without further investigation across other datasets, but it is likely that the increased complexity of the FFHQ

dataset compared to LSUN Churches is what results in the decreased impact of the normed ELBO. This is as the weighting of the normed ELBO is similar to the standard ELBO, while the reweighted ELBO is significantly different and forces the models to train more precisely, improving performance on more-complex datasets, while on simpler datasets changing the normalisation is sufficient to see improvements in validation ELBO.

6.3 Comparison to Autoregressive Models

As described in Sec. 4.3.1, autoregressive techniques are the most common approach for training transformers such as the ones used in the MLM Diffusion models. As such, it is important to include a comparison of these two techniques to demonstrate that the improvements in sample quality are not only from architectural improvements but also from the training methods used.

6.3.1 Negative Log-Likelihood

Autoregressive models are trained directly on the negative log-likelihood (NLL) through the utilisation of the chain rule. This is in contrast to the game-theory inspired methods used to train GANs [19]. While both methods have their advantages, directly optimising the NLL is simpler to implement and debug than adversarial approaches such as GANs, as well as offering more stable training. It is possible to approximate the likelihood of the MLM Diffusion models through the use of an additional layer added to the generator of the VQGAN, enabling a direct comparison of NLL on both techniques.

The probabilistic layer of the generator is trained after the VQGAN has already been trained to convergence. This layer takes the final reconstructions from the generator and approximates values for μ and σ, in a similar approach to that used by Variational Autoencoders, where $p(x|z)$ is assumed to be normally distributed. Using these probabilistic values it is possible to compute $p(x|z)$, and $p(z)$ can be approximated by computing the ELBO of the compressed latents using a trained MLM Diffusion sampler. As $p(x) = p(x|z)p(z)$, it is now simple to compute an approximation of $p(x)$ for the MLM Diffusion models, without the use of autoregressive techniques.

6.3.2 Results

The results for the comparison between the autoregressive models and MLM Diffusion can be seen in Tab. 4. MLM Diffusion outperforms the autoregressive models on both metrics across both datasets, demonstrating the increased generative capability of the new models.

It is important to note that the autoregressive models were trained without any form of regularisation, such as weight decay or dropout. While these regularisation techniques will delay or even prevent overfitting in the autoregressive models and thus improve NLL, they negatively impact sample quality and increase FID. As such, the approach with the lowest FID (i.e. with no regularisation) was selected for these comparisons.

Method	Churches		FFHQ	
	FID ↓	NLL ↓	FID ↓	NLL ↓
Autoregressive	5.93	6.24	8.15	6.18
MLM Diffusion	**5.58**	**6.01**	**7.12**	**5.96**

Table 4: FID and validation NLL (in BPD) for different methods to approximate discrete latents using the same Transformer architecture on LSUN Churches and FFHQ. Trained for 1M steps, with near-identical training times due to an equal number of forward and backward passes on an identical architecture.

6.4 Sampling Speed

One of the primary advantages of training a diffusion model compared to many other generative models is that it is possible to trade off sample quality for increased sampling speed, without the need to retrain the model. This section analyses the effect that increasing sampling speed has on sample quality of images generated by MLM Diffusion models.

6.4.1 Reducing sampling steps

Using fewer sampling steps than those used during training is very simple: rather than unmasking one token at a time, which results in 256 sampling steps for 16×16 latents, multiple tokens are unmasked at each sample step. For example, if sampling 4 tokens per step, it would take only 64 sampling steps to fully-unmask the sample image. This effectively results in skipping timesteps during the diffusion sampling process, as done in continuous DDPMs.

There exists a number of schemes for determining how many time steps to skip per sampling step. These including the simple choice of skipping a fixed number of steps, scaling quadratically, or even determining the optimal steps to skip to using dynamic programming [64, 75]. For all the below experiments linear skipping was used as no noteworthy improvements were obtained using any other step-skipping scheme.

6.4.2 Results

The effect of increasing sampling speed on sample quality can be seen in Tab. 5. While there clearly is a trade off between FID and sampling time, models on all 3 datasets still achieve high-quality FID scores even when using only 50 sampling steps, scores that only a few years ago would have been state-of-the-art. This suggests that the MLM Diffusion models are trained well across all time steps and are capable of producing samples in significantly fewer sampling steps than other diffusion models while only slightly decreasing sample quality.

Steps	50	100	150	200	256
Churches	6.86	6.09	5.81	5.68	5.58
Bedroom	6.85	5.83	5.53	5.32	5.42
FFHQ	9.60	7.90	7.53	7.52	7.12

Table 5: FID for different number of sampling steps on LSUN Churches, Bedroom and FFHQ. Diffusion steps are evenly spaced.

6.5 Super-Resolution Image Generation

As detailed in Sec. 5.3.4, the MLM diffusion models are capable of generating images at resolutions greater than those of the training dataset. Unfortunately, there are no quantitative metrics that can be used to analyse these super-resolution samples. Instead, the author has opted to include a selection of samples to demonstrate the efficacy of this technique. All samples were generated using a temperature $t = 0.8$ in order to create higher-fidelity images at the expense of diversity.

6.5.1 Results

As can be seen in Fig. 14 (below) and Fig. 1, the MLM diffusion models are capable of generating of images at resolutions significantly larger than those used during training time. The example images specifically demonstrate generated images of resolution 512×256 or 256×512 pixels, when all training images used were only 256×256

Figure 14: Super-resolution LSUN Churches samples generated using MLM Diffusion models.

6.6 VQGAN Improvements

This section details the results of experiments carried out to evaluate the effect of the VQGAN improvements proposed in Sec. 5.2.2. To evaluate the ability of the VQGAN to reconstruct the original image from the compressed, discretised latent space, the FID metric is used. While this metric is usually used to evaluate the quality of new, unseen, generated samples, it also is an effective metric for evaluating reconstruction quality.

6.6.1 Results

Results from the VQGAN changes experiments can be seen in Tab. 6. By simultaneously applying differentiable augmentations (DiffAug) and limiting the adaptive weighting of the discriminator to a value of 1, the reconstruction quality of the VQGAN improves on both the LSUN Churches and FFHQ datasets.

Modifications	Churches	FFHQ
Default	5.25	3.37
$\lambda_{\max} = 1$	8.67	4.72
DiffAug	5.16	6.57
Both	**2.70**	**3.12**

Table 6: Effect of DiffAug and adaptive weight limiting on reconstruction FID. Results were calculated on Vector-Quantized image models trained on LSUN Churches and FFHQ for 500k steps using a batch size of 5.

However, either of these adaptations applied individually results in poorer quality reconstructions for FFHQ. Furthermore, on LSUN churches, limiting the discriminator weight significantly reduces reconstruction quality (FID 5.25 increasing to 8.67) and DiffAug alone makes a very minimal improvement to FID.

These results are intriguing, as they demonstrate that while differentiable augmentations alone can be effective in improving discriminator performance, they can also introduce instability into the model which results in decreased image quality. By limiting the discriminator's adaptive weighting to a small, fixed maximum, this instability is reduced while preserving the benefits on DiffAug, resulting in increased reconstruction performance and training stability on VQGAN.

Figure 15: Non-cherry picked, $t = 0.85$, 256x256 FFHQ samples.

Figure 16: Non-cherry picked, $t = 0.9$, 256x256 LSUN Bedroom samples.

Limitations

While MLM Diffusion models do achieve competitive and SOTA results on various metrics and have been demonstrated throughout this thesis to be effective high-resolution generative models, they are not without their limitations. This section focuses on highlighting these limitations, while potential avenues for improving the models is discussed in Sec. 8

Training Time and Scalability All models trained for the experiments in this paper were trained to generate 256×256 pixel images. This is an impressive feat and would have been unimaginable as little as 6 years ago. However, in comparison to the size of images used most commonly on the internet these generated images are still relatively low-resolution.

While it is possible to scale MLM diffusion models to larger resolutions, this comes with a significant cost. To gather the results presented in this thesis, a VQGAN was first trained for approximately 2 weeks in order to be able to effectively compress images from the chosen dataset into a discrete, 16×16 latent space. The VQGANs were trained using a very small batch size of 5 in order to meet the 11GB RAM limitations of the GPUs used. For the second training stage, MLM Diffusion samplers were trained for between 4 and 7 days on this compressed latent space before achieving convergence. This results in an approximate total training time of 3 weeks on a single NVIDIA 2080ti when using 256×256 images.

If one were to build models to train on larger datasets, VQGANs with many more parameters and larger latent spaces would need to be trained for the first training stage, and this training time would scale quadratically with respect to the image resolution, at best. Furthermore, as the larger VQGANs would use many more parameters, GPUs with significantly more RAM or multiple GPUs would need to be used for training them. This may be feasible for large deep-learning research centres but it would make it highly impractical for day-to-day usage or for an amateur practitioner with limited computational resources. While not as limited with respect to memory, the training time for the MLM Diffusion samplers also scales quadratically with the resolution of the latent space, which gives an optimistic minimum of 28 days training time for the 32×32 latent spaces that would be appropriate for 1024×1024 images.

Failed Super-Resolution Generation on FFHQ The other primary limitation of the methods discussed in this thesis is the ineffectiveness of the super-resolution method (Sec. 5.3.4) on generating larger images on the FFHQ dataset.

The structure of the FFHQ dataset does not lend itself to naturally being elongated in either the horizontal or vertical direction in the same way as the LSUN datasets. Unlike in LSUN Bedrooms or Churches, the focus of each image in FFHQ is the centre of the image, with the boundaries of the image being significantly less important when it comes to reconstruction and sample quality. This means that striding along the image in order to collect local subsets $c_j(\bar{z}_t)$ to elongate the latent space along its boundaries results in poor quality samples, as the VQGAN and MLM Diffusion samplers have learned to focus the majority of the key information in the centre of the image.

Conclusion

Summary This thesis introduces a novel method, named MLM Diffusion, for generating high-resolution images from discrete latent spaces. This technique utilises bidirectional transformers in a masked diffusion process to generate diverse, high-quality and globally consistent images from a single-point latent space. Experiments carried out on these models demonstrate they are capable of achieving state-of-the-art results on PRDC scores for a number of high-resolution datasets, as well as performing competitively with adversarial models on FID. These models only require a single NVIDIA 2080ti GPU with 11GB of RAM for training on 256×256 datasets and can be trained and sampled from in only a fraction of the time required for continuous DDPMs trained on the data space.

The introduced models offer a number of benefits over similar existing methods. Unlike GANs, these models do not suffer from mode-collapse and are much more stable during training while still outperforming them on a number of quantitative metrics. Furthermore, MLM Diffusion allow for improved consistency and significantly-reduced overfitting when compared to the autoregressive approaches traditionally used for this task. This is as using a diffusion process allows the sampler to access the whole latent space during the sampling process, whereas autoregressive models are trained on a fixed, user-defined token ordering that constrains the scope of the generative process. MLM Diffusion models also allow one to approximate the likelihood $p(x)$ of a given sample, overcoming the major flaw of adversarial models, which are not capable of producing such estimations. Also, as MLM Diffusion models use a diffusion process for training and sampling, they have the flexibility to trade off sampling time for generated-image quality, allowing images to be generated in half as many steps as used during training, with a near-negligible reduction in FID.

Finally, this thesis introduces improvements for the VQGAN models used to produce discrete, compressed latent representations of the target datasets. This allows for higher-quality and more diverse samples to be generated from such latent spaces.

Future Work There are many areas that could be explored to build upon the work presented in this thesis. Firstly, training on other high-resolution image datasets and using class-conditional training would potentially highlight other strengths and weaknesses of this method. It would be also be incredibly interesting to see this technique applied in non-image domains: text data is often represented in a discrete format and would potentially respond very well to diffusion techniques for text generation and prediction.

Further work is needed to enable these models to scale and generate even higher-resolution images. This may be as simple as training using multiple, more-powerful GPUs, which would at the very least give more precise insight on the time and memory requirements necessary when scaling to larger images. A key bottleneck in the scaling of this technique is the VQGAN used for compression and discretisation. Thorough investigation of other discretisation methods such as gumbel-softmax, [31] or scaling techniques such as progressive growing [34] may yield substantial improvements in training and sampling speed on higher-resolution images.

References

[1] Serguei Barannikov, Ilya Trofimov, Grigorii Sotnikov, Ekaterina Trimbach, Alexander Korotin, Alexander Filippov, and Evgeny Burnaev. Manifold Topology Divergence: a Framework for Comparing Data Manifolds. *arXiv preprint arXiv:2106.04024*, 2021. 27

[2] Yoshua Bengio. Estimating or propagating gradients through stochastic neurons, 2013. 8

[3] Sam Bond-Taylor, Peter Hessey, Hiroshi Sasaki, Toby P. Breckon, and Chris G. Willcocks. Unleashing transformers: Parallel token prediction with discrete absorbing diffusion for fast high-resolution image generation from vector-quantized codes. *CoRR*, abs/2111.12701, 2021. 3, 18

[4] Sam Bond-Taylor, Adam Leach, Yang Long, and Chris G Willcocks. Deep Generative Modelling: A Comparative Review of VAEs, GANs, Normalizing Flows, Energy-Based and Autoregressive Models. *arXiv preprint arXiv:2103.04922*, 2021. 4, 12

[5] Ali Borji. Pros and Cons of GAN Evaluation Measures: New Developments. *arXiv preprint arXiv:2103.09396*, 2021. 27

[6] 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

[7] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020. 13, 25

[8] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T. Freeman. Maskgit: Masked generative image transformer, 2022. 18

[9] Xuelin Chen, Daniel Cohen-Or, Baoquan Chen, and Niloy J Mitra. Towards a Neural Graphics Pipeline for Controllable Image Generation. In *Computer Graphics Forum*, volume 40, pages 127–140. Wiley Online Library, 2021. 4

[10] Rewon Child. Very Deep VAEs Generalize Autoregressive Models and Can Outperform Them on Images. In *International Conference on Learning Representations*, 2021. 4, 6, 7, 8, 28, 29

[11] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating Long Sequences with Sparse Transformers. *arXiv:1904.10509*, 2019. 4, 11

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*, 2019. 14, 43

[13] Sander Dieleman, Aäron van den Oord, and Karen Simonyan. The Challenge of Realistic Music Generation: Modelling Raw Audio at Scale. In *Advances in Neural Information Processing Systems*, volume 31, 2018. 8

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020. 18

[15] Yilun Du and Igor Mordatch. Implicit Generation and Generalization in Energy-Based Models. In *Advances in Neural Information Processing Systems*, volume 33, 2019. 9, 10

[16] Patrick Esser, Robin Rombach, Andreas Blattmann, and Björn Ommer. Imagebart: Bidirectional Context with Multinomial Diffusion for Autoregressive Image Synthesis. *arXiv preprint arXiv:2108.08827*, 2021. 18, 28, 43

[17] Patrick Esser, Robin Rombach, and Björn Ommer. Taming Transformers for High-Resolution Image Synthesis. *arXiv:2012.09841*, 2021. 8, 9, 11, 18, 19, 28, 29, 43, 44

[18] Lukas Fetty, Mikael Bylund, Peter Kuess, Gerd Heilemann, Tufve Nyholm, Dietmar Georg, and Tommy Löfstedt. Latent Space Manipulation for High-Resolution Medical Image Synthesis via the StyleGAN. *Zeitschrift für Medizinische Physik*, 30(4):305–314, 2020. 4

[19] 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*, volume 27, 2014. 4, 8, 9, 15, 18, 31

[20] Will Grathwohl, Kevin Swersky, Milad Hashemi, David Duvenaud, and Chris J Maddison. Oops I Took A Gradient: Scalable Sampling for Discrete Distributions. In *International Conference on Machine Learning*, 2021. 10

[21] Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. *CoRR*, abs/1912.03263, 2019. 10

[22] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Günter Klambauer, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a nash equilibrium. *CoRR*, abs/1706.08500, 2017. 26

[23] Geoffrey E. Hinton. Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, 2002. 9

[24] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, volume 33, 2020. 4, 15, 18, 28

[25] Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded Diffusion Models for High Fidelity Image Generation. *arXiv preprint arXiv:2106.15282*, 2021. 4, 5, 18

[26] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997. 13

[27] 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

[28] Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 1933. 6

[29] Drew A Hudson and C. Lawrence Zitnick. Generative Adversarial Transformers. *Proceedings of the 38th International Conference on Machine Learning, ICML*, 2021. 28

[30] Phillip Isole, Jun-Yan Zhu, Zhou Tinghui, and Alexei A. Efros. Image-to-Image Translation with Conditional Adversarial Networks. In *CVPR*, 2017. 9

[31] Eric Jang, Shixiang Gu, and Ben Poole. Categorical Reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations*, 2017. 37

[32] Heewoo Jun, Rewon Child, Mark Chen, John Schulman, Aditya Ramesh, Alec Radford, and Ilya Sutskever. Distribution Augmentation for Generative Modeling. In *ICML*, 2020. 11, 20

[33] Andrej Kaparthy. MinGPT: A PyTorch re-implementation of GPT training. *https://github.com/karpathy/minGPT*. 25

[34] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive Growing of GANs for Improved Quality, Stability, and Variation. In *International Conference on Learning Representations*, 2018. 4, 18, 28, 29, 37

[35] Tero Karras, Samuli Laine, and Timo Aila. A Style-Based Generator Architecture for Generative Adversarial Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019. 18, 27, 29, 44

[36] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and Improving the Image Quality of StyleGAN. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 28, 29

[37] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations*, 2014. 4, 6, 18

[38] Solomon Kullback. Letter to the editor: The kullback-leibler distance. *The American Statistician*, 1987. 6

[39] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved Precision and Recall Metric for Assessing Generative Models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, 2019. 27

[40] Petro Liashchynskyi and Pavlo Liashchynskyi. Grid search, random search, genetic algorithm: A big comparison for NAS. *CoRR*, abs/1912.06059, 2019. 44

[41] Ji Lin, Richard Zhang, Frieder Ganz, Song Han, and Jun-Yan Zhu. Anycost GANs for Interactive Image Synthesis and Editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14986–14996, 2021. 4

[42] Bingchen Liu, Yizhe Zhu, Kunpeng Song, and Ahmed Elgammal. Towards Faster and Stabilized GAN Training for High-fidelity Few-shot Image Synthesis. In *International Conference on Learning Representations*, 2021. 4

[43] Jacob Menick and Nal Kalchbrenner. Generating High Fidelity Images with Subscale Pixel Networks and Multidimensional Upscaling. In *International Conference on Learning Representations*, 2019. 4

[44] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 1953. 10

[45] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral Normalization for Generative Adversarial Networks. In *International Conference on Learning Representations*, 2018. 4

[46] Muhammad Ferjad Naeem, Seong Joon Oh, Youngjung Uh, Yunjey Choi, and Jaejun Yoo. Reliable Fidelity and Diversity Metrics for Generative Models. In *International Conference on Machine Learning*, pages 7176–7185, 2020. 27

[47] Charlie Nash, Jacob Menick, Sander Dieleman, and Peter W Battaglia. Generating Images with Sparse Representations. *arXiv preprint arXiv:2103.03841*, 2021. 28, 29, 44

[48] Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. *CoRR*, abs/2102.09672, 2021. 15

[49] Anton Obukhov, Maximilian Seitzer, Po-Wei Wu, Semen Zhydenko, Jonathan Kyl, and Elvis Yu-Jing Lin. High-fidelity performance metrics for generative models in pytorch, 2020. Version: 0.3.0, DOI: 10.5281/zenodo.4957738. 29

[50] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image Transformer. In *ICML*, 2018. 4, 11, 13

[51] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualised word representations. *NAACL*, 2018. 13

[52] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018. 13

[53] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019. 13

[54] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-Shot Text-to-Image Generation. *arXiv preprint arXiv:2102.12092*, 2021. 8

[55] Ali Razavi, Aaron van den Oord, and Oriol Vinyals. Generating Diverse High-Fidelity Images with VQ-VAE-2. *NeurIPS 32*, 2019. 7, 8, 28, 29

[56] Scott Reed, Aäron Oord, Nal Kalchbrenner, Sergio Gómez Colmenarejo, Ziyu Wang, Yutian Chen, Dan Belov, and Nando Freitas. Parallel Multiscale Autoregressive Density Estimation. In *International Conference on Machine Learning*, 2017. 4

[57] Danilo Jimenez Rezende and Shakir Mohamed. Variational Inference with Normalizing Flows. *ICML*, 2015. 15

[58] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015. 16

[59] Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. Efficient Content-Based Sparse Attention with Routing Transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68, 2021. 11

[60] Mehdi SM Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing Generative Models via Precision and Recall. In *Advances in Neural Information Processing Systems*, volume 31, 2018. 27

[61] Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P. Kingma. PixelCNN++: Improving the PixelCNN with Discretized Logistic Mixture Likelihood and Other Modifications. *ICLR*, 2017. 4, 11

[62] Hiroshi Sasaki, Chris G Willcocks, and Toby P Breckon. UNIT-DDPM: UNpaired Image Translation with Denoising Diffusion Probabilistic Models. *arXiv preprint arXiv:2104.05358*, 2021. 4

[63] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. In *International Conference on Machine Learning*, 2015. 23

[64] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *CoRR*, abs/2010.02502, 2020. 4, 15, 16, 32

[65] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *CoRR*, abs/1907.05600, 2019. 15

[66] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-Based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations*, 2021. 4, 18

[67] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014. 26

[68] Anton Tsitsulin, Marina Munkhoeva, Davide Mottin, Panagiotis Karras, Alex Bronstein, Ivan Oseledets, and Emmanuel Müller. The Shape of Data: Intrinsic Distance for Data Distributions. In *International Conference on Learning Representations*, 2020. 26, 27

[69] Arash Vahdat and Jan Kautz. NVAE: A Deep Hierarchical Variational Autoencoder. In *Advances in Neural Information Processing Systems*, volume 33, 2020. 6

[70] Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based Generative Modeling in Latent Space. *arXiv preprint arXiv:2106.05931*, 2021. 4

[71] Aaron van den Oord, Nal Kalchbrenner, Lasse Espeholt, koray kavukcuoglu, Oriol Vinyals, and Alex Graves. Conditional Image Generation with PixelCNN Decoders. *NeurIPS 29*, 2016. 11

[72] Aäron Van Den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel Recurrent Neural Networks. In *ICML*, 2016. 4, 11

[73] Aaron van den Oord, Oriol Vinyals, and koray kavukcuoglu. Neural Discrete Representation Learning. *NeurIPS 30*, 2017. 4, 7, 11, 18

[74] 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*, volume 30, 2017. 4, 11, 12, 14, 18

[75] Daniel Watson, Jonathan Ho, Mohammad Norouzi, and William Chan. Learning to efficiently sample from diffusion probabilistic models. *CoRR*, abs/2106.03802, 2021. 16, 32

[76] Zhisheng Xiao, Karsten Kreis, Jan Kautz, and Arash Vahdat. VAEBM: A Symbiosis between Variational Autoencoders and Energy-based Models. In *International Conference on Learning Representations*, 2021. 4, 10

[77] Ning Yu, Connelly Barnes, Eli Shechtman, Sohrab Amirghodsi, and Michal Lukac. Texture Mixer: A Network for Controllable Synthesis and Interpolation of Texture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12164–12173, 2019. 4

[78] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable Augmentation for Data-Efficient GAN Training. In *Advances in Neural Information Processing Systems*, volume 33, 2020. 20

Appendix

9.1 Experiment Hyperparameters

Hyperparameter	All Datasets
Base Learning Rate	4.5e-6
β Eq. (26)	0.25
Differential Augmentation	False
Attention Resolution	[16]
Batch Size	3
Channel Multipliers	[1, 1, 2, 2, 4]
Codebook Size	1024
Disc Layers	3
Discriminator Weight Max Eq. (25)	1
Discriminator Start Step	30001
Embedding Dimension	256
Latent Shape	[1, 16, 16]
Discriminator Hidden Features	64
AE Hidden Features	128
Perceptual Weight	1.0

Table 7: VQGAN Hyperparameters (same for all datasets)

Hyperparameter	LSUN	FFHQ
Dropout	0.0	0.0
Transformer Embedding Dimension	512	512
No. Transformer Heads	8	8
No. Transformer Layers	24	24
Block Size	256	256
Temperature	1.0	1.0
Batch Size	20	20
Learning Rate	2e-4	1e-4
Warmup Iterations	10000	30000

Table 8: MLM Diffusion Hyperparameters

9.2 Hyperparameter Tuning

In the present study, the hyperparameters delineated in Tab. 7 were predominantly set to mirror the default values established by [17]. However, changes to these defaults were made, including the incorporation of the Discriminator Weight Max variable within the VQGAN framework and minor modifications to other variables for specific reasons, such as batch size reduction to reduce memory footprint and learning rate reduction to improve training stability.

Regarding the hyperparameters delineated in Tab. 8, the selection process was similar to the previous one: it initially entailed determining reasonable default values for transformers based on the most recent literature (e.g. [12], [17], [16]) at the time and considering the constraints

imposed by memory requirements. Certain parameters were adjusted manually to enable stable training, such as learning rate and warmup iterations. Due to the long training duration associated with these models and the fact that they were already performing at SOTA, architectural parameters such as number of heads of layers were left unchanged.

It is likely that an automated Neural Architecture Search [40] approach to optimising the experiment hyperparameters would yield slightly-improved image generation performance. However, it is our suspicion that these improvements would be negligible in comparison with the time and resources necessary to investigate this task. The works compared to MLM Diffusion in this paper, [35, 17, 47], adopt a similar approach, opting for practicality over precision when it comes to tuning hyperparameters. No other works report using automated approaches for hyperparameter tuning and it would be unlikely to significantly improve their results.