MEGAPIXEL IMAGE GENERATION WITH STEP-UNROLLED DENOISING AUTOENCODERS

Alex F. McKinney & Chris G. Willcocks
Department of Computer Science
Durham University
Durham, UK
{alexander.f.mckinney,christopher.g.willcocks}@durham.ac.uk

ABSTRACT

Recent research has pushed sample resolutions higher whilst reducing computational requirements and sampling speeds. One approach is to utilize powerful vector-quantization (VQ) models to reduce computational requirements whilst still producing high quality samples. In this work, we push this further through the use of non-autoregressive (NAR) denoising autoencoders (SUNDAE) and modifications to hierarchical transformers that have found recent success in language modelling. This approach to allows for very fast sampling and training (4-6 days) of VQ latents from pre-trained VQ-GAN models. Furthermore, we found the NAR nature of the model made it suitable for complex inpainting with arbitrary masks. Finally, we trained a new VQ-GAN model on a dataset of faces at resolutions exceeding one million pixels, ultimately allowing allowing for megapixel image generation in only two seconds on consumer-grade GPUs.

Figure 1: High-resolution samples produced using our non-autoregressive approach. Each 1024×1024 sample was generated in $\approx$ 2 seconds on a GTX 1080Ti – including both discrete latent sampling and subsequent decoding. The SUNDAE sampler was trained in 4 days on a single V100 32GB.

1 INTRODUCTION

An ideal deep generative model would satisfy three key requirements: high-quality samples, sample diversity via mode coverage, and computational inexpensive sampling. Arguably, there are other desirable properties such as a meaningful latent space and exact likelihood calculation, however no current generative model can satisfy all three requirements – let alone additional attractive properties – thus forming the so-called generative modelling trilemma [41].

Models such as generative adversarial networks (GANs) [13] excel at high-quality and fast sampling, but often fail to model the entire data distribution due to not directly optimising for likelihood – using an adversarial loss as a proxy. Variational auto-encoders (VAEs) [19] offer excellent mode coverage and fast sampling speeds, but the resulting samples are often blurry even at small resolutions, and have little hope of scaling to greater resolutions like GANs.

Autoregressive (AR) models such as PixelSnail [5], Image Transformer [23], and DALL-E [23] have demonstrated respectable sample quality and mode coverage, even including zero-shot image generation [25]. However, they are computationally expensive to sample from, requiring many

network iterations, making them infeasible for interactive applications. Denoising diffusion probabilistic models (DDPMs) [14] and score-based models (SBMs) [29, 30, 31] produce samples that rival or even exceed the quality of GANs [11] whilst still providing good mode coverage, but are still plagued by potentially requiring thousands of network evaluations.

Vector-quantized image modelling [37, 26, 12] alleviates sampling speed issues in AR methods by reducing the spatial dimension at which AR sampling occurs. This results in excellent quality samples whilst improving sampling speeds, but often requires a two-stage approach and still does not match the speed of GANs. Recent work has applied vector-quantization (VQ) methods to diffusion models [3] allowing for fast parallel decoding. Other work does not use vector-quantized spaces, but latent spaces nonetheless, to accelerate sampling [41, 34].

From this brief overview of generative modelling literature, it is clear that indeed no single model satisfies all three conditions. This therefore motivates research into explicitly addressing this trilemma. In this work, we move towards such a solution, beginning from existing work applying generative models to discrete latents. This provides an excellent starting point in terms of sample quality and a respectable amount of mode coverage, but an unacceptably slow sampling speed despite the reduced spatial dimension of the discrete latent space. We directly address this issue by instead sampling discrete latents using modern non-autoregressive (NAR) generative models in an effort to close the gap, sampling speed wise, with constant iteration complexity models such as GANs. Specifically, we use discrete step-unrolled denoising autoencoders (SUNDAE) [28] to gradually denoise samples from a uniform prior into one that matches the prior over the discrete latent space defined by a pre-trained vector-quantized GAN (VQ-GAN) model. This forms our first research question, whether SUNDAE models is suitable paradigm for sampling discrete latent codes. As a result of our studies, we find that indeed they are highly suited for this task.

These discrete SUNDAE models have only previously be applied to language modelling tasks [28]. In their work, they used transformer [39] architectures to form their autoencoder. In parallel, a noticeably more efficient variant of transformers – the Hourglass Transformer [22] – was released, which has a hierarchical architecture aimed at language and image modelling tasks, though potentially flawed on the latter. This raises a second research question, whether the approach introduced in Savinov et al. [28] is also suited for latent modelling, and whether the hierarchical approach introduced in Nawrot et al. [22] can be further improved upon. As a result of our studies, we find that although they are suited for latent modelling, there are many modifications we can make to improve their performance further.

Given a fast sampling and an efficient transformer architecture, we are potentially left with a highly scaleable (with respect to spatial resolution) model. This raises a third and final research question, whether our new approach allows for the generation of extremely high resolution images in a time frame that would permit fully-, or near-interactive use. Again, as a result of our studies, we find that this is indeed the case, proven by the synthesis of 1024×1024 images in as few as 2 seconds on a consumer-grade GPU. To our knowledge, this is the fastest sampling model at this resolution, with the exception of pure GAN-based approaches. This required the training of our own VQ-GAN model, which was, also to our knowledge, the largest VQ-GAN model trained in terms of input size.

Our project objectives were as follows:

- Apply SUNDAE to the task of VQ latent generation and ultimately image generation, and perform analysis into their advantages of being used for this task over existing approaches, including both AR and NAR solutions.
- Analyse the methods used in Savinov et al. [28] and Nawrot et al. [22] and design modifications that make these approaches more suitable for VQ latent modelling.
- Attempt to scale our model to extremely high resolution images and analyse successes and failures when operating at such resolutions.

2 Related Work

This work builds upon much prior research into powerful deep generative models, self-supervised methods, and efficient transformer architectures. We briefly cover relevant prior work into deep generative models in general in §2.1-2.3, a particular NAR generative model we wish to build upon

in §2.4, and a recent and highly effective development into a efficient transformer architecture in §2.5.

2.1 Autoregressive Generative Models

One major deep generative model family is autoregressive (AR) models, characterised by a training and inference process based on the probabilistic chain rule. During training, they directly aim to maximise the likelihood of the data they are trained on, which leads to excellent mode coverage. Prior work using these methods resulted in impressive results in terms of both sample quality and diversity, but are ultimately unwieldy for use in real world applications due to their slow sampling speed.

The slow sampling speed is due to their sequential nature, defined by the chain rule of probability. Given an input $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$, an AR model $p_\theta(\cdot)$ can generates new samples sequentially:

$$p_\theta(\mathbf{x}) = p_\theta(x_1, \ldots, x_n) = \prod_{i=1}^{n} p_\theta(x_i | x_1, \ldots, x_{i-1}) \tag{1}$$

meaning that the number of sampling steps is equal to the size of the decomposition of $\mathbf{x}$, making this slow for large inputs.

For certain tasks, the ordering of the decomposition of $\mathbf{x}$ is obvious, for example on text or speech. For images this is less obvious, however typically a raster scan ordering is used. Certain AR models are order-agnostic, allow for arbitrary ordering to be used during training and inference.

One class of AR models are recurrent neural networks (RNNs) which are an early example of using neural networks to model sequential data, such as text, audio, time-series data, or even vector handwriting strokes. Though they can be used as purely classification or regression models, they are also suited for use as generative models by modelling the relationship shown in Equation 1. They do suffer from a number of issues, most notably vanishing gradients [24] and inability to model long-range relationships between items in the input. Long short-term memory recurrent neural networks (LSTMs) [15] improved upon them further by introducing dedicated memory units, allowing for the modelling of longer range relationships. Later gated recurrent units (GRUs) [7] simplified the architecture whilst still retaining good performance. With the advent of transformer architectures [39], modelling even longer relationships became possible, even at a full-document level. It also intro cued the capability to train on all sequence elements in parallel through the use of causal masking, therefore not violating the autoregressive property.

Applying AR models to images followed a similar trend. PixelRNN [35] used two-dimensional recurrent layers and residual connections to model the distribution of raw pixel values. The same paper also introduced PixelCNN, which it claims had worse performance but were faster to train. These were extended to allow for conditional generation in [36]. Later work augmented PixelCNN with self-attention mechanism, forming PixelSnail [5], which can therefore model longer relationships than a fully convolutional or recurrent architecture. Image Transformer [23] later applied transformer architectures to the same task through an effective but altogether conceptually simple approach.

2.2 Non-autoregressive Generative Models

Non-autoregressive (NAR) generative models include GANs, SBMs and DDPMs, flow-based models, energy-based models (EBMs), and implicit models. Though the number of sampling steps is now independent of the data dimensionality (as we no longer use the chain rule of probability to sample) the actual number of steps varies greatly: from single-step generation in GANs to potentially many thousands in the original diffusion model literature.

Perhaps the most infamous class of NAR generative model – or perhaps generative model overall – are generative adversarial networks (GANs) [13]. These typically consist of two components: a generator that create images from some latent variable (in the most basic case just random noise), and a discriminator that tries to distinguish images from the dataset from images generated by the generator network [13]. They are known for high-fidelity samples, fast sampling, unstable training, and tendency to collapse onto certain modes of the underlying distribution due to not optimising directly

for likelihood. This is reflected in its relatively low-diversity samples. Nonetheless, the quality of the samples has made them a popular choice in a variety of applications, including unconditional and conditional generation [17, 4] image, audio synthesis [21], style transfer [44]. They can even be applied to discrete data [9], but are less effective on such domains due to the non-differentiability of discrete samples.

Another class of generative model are variational auto-encoders (VAEs) [19] which allow for sampling in a single forward pass like GANs, but are trained to directly maximise likelihood. Specifically, VAEs map inputs to latent variables that follow some easy to sample from, but still sufficiently complex, prior distribution. A common choice is a multivariate Gaussian with diagonal covariance [19]. A decoder network maps these latent codes back to the data distribution. Although this approach is successful on small datasets, on more complex datasets the samples and reconstructions tend to become blurry, suggesting a simple prior is unable to perfectly fit the distribution. Later work extended VAEs to be hierarchical, having multiple Gaussian priors [33, 6] which were found to outperform purely autoregressive models.

Removing the causal constraints also allows for bidirectional context during sampling and flexible inpainting patterns, rather than being limited to left-to-right inpainting in autoregressive models.

2.3 Vector Quantized Image Modelling

Learning useful representations, also known as latent codes, in an unsupervised manner is a key challenge in machine learning. Historically, these representations have been in a continuous form, but in more recent literature they are often discrete. An early example of this is vector-quantized VAE (VQ-VAE) [37], a variant on VAE representation models. A VQ-VAE has three main components: an encoder network, a codebook, and a decoder. The encoder network outputs a compressed representation of the input, and the codebook $\mathcal{C}$ quantizes these representations, outputting a discretized representation of indices from 1 to the codebook size v. Each index i maps to one of the codebook embeddings e_i. The decoder then maps the quantized embeddings back to the original signal, training it in tandem to reconstruct the input signal and to minimize additional codebook loss terms [37]. Once a trained VQ-VAE model has been produced, a powerful auxiliary generative model can be trained to generate these discrete latent representations and then the decoder can produce the final sample. In the original work on VQ-VAE, they use a PixelCNN model to generate the discrete latent codes [37], though any generative model on discrete data can be used.

Later approaches extended VQ-VAE to multiple distinct codebooks in VQ-VAE-2 [26]. Though theoretically it can be extended to any number of codebooks, they performed experiments on a two-level and three-level model, applying the latter to 1024×1024 images. They then sampled the resulting combined discrete codes use PixelSnail [5], each code conditioned on all previous levels in the hierarchy. Though faster to sample than applying a generative directly to pixels, at such a resolution and with multiple levels to sample, sampling times were still slow.

Aside from sampling autoregressively, a reason for the slow sampling speed was the large spatial resolution of the discrete codes. Albeit smaller than the original signal, VQ-VAE is limited in how much it can compress the signal via a simple reconstruction objective before it loses too much perceptual quality due to the rate-distortion trade-off. For example, the original VQ-VAE only has a downsampling rate of $f = 4$ [37] and VQ-VAE-2 has a rate of top-level rate of $f = 32$ but requires a total of three discrete latent codes in order to achieve this [26]. By introducing perceptual and adversarial loss terms, VQ-GAN [12] is able to achieve compression rates of $f = 16 \sim 32$ with only a single discrete latent representation whilst retaining high quality reconstructions [12]. The greater weighting on perceptual and adversarial loss does mean, however, that VQ-GAN does sometimes edit its reconstructions, rather than attempt to preserve all details perfectly like when using a simple reconstruction loss alone. Later, improvements such as differential data augmentation [3], codebook improvements, and a transformer-based architecture [42] improved reconstruction quality further.

Originally designed for audio compression [43], Residual VQ proposes the use of multiple codebooks to recursively quantize and refine the residual of an input signal. This produces multiple discrete representations, which can later be reconstructed by the decoder to decompress the waveform. Additionally, individual codebooks can be dropped out, allowing for variable bit-rates [43]. Concurrently to this work, Lee et al. [20] used residual VQ to represent images with a compression ratio of $f = 32$ and then trained a transformer model to autoregressively predict the stack of discrete

tokens at a given spatial location, allowing for fast sampling despite actually having multiple levels of discrete latent representations[20].

A typical strategy for selecting which codebook vector e_i to map to a particular input is to compute the Euclidean distance between a given continuous input and the codebook centroid, and then pick the $\arg\min$ [37]. This is denoted the k-means strategy. This strategy does result in a phenomena known as codebook collapse, where certain codebook vectors never get used, harming the downstream reconstruction quality. An alternative method is to use the Gumbel-Softmax [16] to select codebook vectors, which typically increases codebook utilisation but often leads to worse reconstruction quality [3].

The issue of codebook collapse is quite significant and there have been a number of attempts to remediate it. [42] found that a lower codeword dimension and codeword normalization improved utilisation. [43] proposed setting a threshold for "stale" codes, and reinitialisating them to a random vector from the current batch when they fall below this threshold. [20] proposed the sharing of a single codebook across many quantizers and additionally stochastically sampled the codes as a function of their distance to the centroid, rather than always taking the $\arg\min$.

All these previous approaches use VQ models to enhance existing AR models, primarily to improve their sampling speed by reducing the spatial dimension we are operating over. Few work directly addresses the discrete prior model itself. Discrete diffusion models [1] are NAR approach to generate discrete data. This was applied to VQ-GAN latents in Bond-Taylor et al. [3], allowing for fast sampling, flexible inpainting, and high fidelity outputs.

2.4 Step-unrolled Denoising Autoencoder

One recent NAR model is SUNDAE [28] which was evaluated on three language modelling tasks: unconditional text-generation, inpainting of Python code, and machine translation – setting a new state-of-the-art among NAR models for the machine translation task [28]. It also demonstrates exceptionally fast sampling, producing high quality samples in as few as 10 steps.

SUNDAE is trained using a denoising objective, akin to the BERT denoising objective [40] but with multiple denoising steps. Given a uniform prior p_0 over some space $Z = \{1, \dots, v\}^N$ where N is the size of the space and v is the vocabulary size, consider the Markov process $\mathbf{z}_t \sim f_\theta(\cdot|\mathbf{z}_{t-1})$ where f_θ is a neural network parameterised by θ, then $\{\mathbf{z}_t\}_t$ forms a Markov chain. This gives a t-step transition function:

$$p_t(\mathbf{z}_t|\mathbf{z}_0) = \sum_{\mathbf{z}_1,\dots,\mathbf{z}_{t-1} \in Z} \prod_{s=1}^{t} f_\theta(\mathbf{z}_s|\mathbf{z}_{s-1}) \tag{2}$$

[28] and, given a constant number of steps T, our model distribution $p_T(\mathbf{z}_T|\mathbf{z}_0)p_0(\mathbf{z}_0)$ – which is clearly intractable.

Instead, they propose an *unrolled denoising* training method that uses a far lower T than is used for sampling [28]. To compensate, they unroll the Markov chain to start from corrupted data produced by a *corruption distribution* $\mathbf{z}' \sim q(\cdot|\mathbf{z})$ rather than from the prior p_0 so the model encounters samples more akin to those seen during the full unroll at sample time [28]. Typically, $T = 2$ during training, as a single step would be similar to the training strategy of BERT [10] but would lead to worse performance as seen in earlier work using BERT as a random field language model [40].

The training objective of SUNDAE is simply the average of all reconstruction losses $L^{(1:T)}(\theta) = \frac{1}{T}\left(L^{(1)}(\theta) + \cdots + L^{(T)}(\theta)\right)$ of the chain after t steps, which is shown to form an upper bound on the actual negative log-likelihood [28]. Taking more steps T leads to a minor improvement in performance, but considerably slows down training time [28] and increases memory usage.

One advantage of this approach is that sampling starts from random tokens, rather than a dedicated "masking" token [3, 1]. Unmasking approaches means that $T \leq N$ as at minimum, one token is unmasked per step. Additionally, it allows the model to be able to "change its mind" about previously predicted positions during sampling, allowing it to make fine-grained adjustments or fix accumulated errors.

2.5 Hourglass Transformers

Vanilla transformers incur a hefty memory and time complexity of $O(L^2)$ for each block [39]. This is largely due to the multi-head self-attention mechanism, as each input position must attend to every other. Most research into efficient transformers focuses on improving the efficiency of these attention mechanism, such as through sparse attention patterns or approximations of attention.

Recent work however, is now focusing on making the overall architecture more efficient. Funnel-Transformer [8] progressively downsamples the input sequence and hence reduces the computational cost of the model. The saved floating point operationss (FLOPs) can then be reassigned to create deeper or wider models and thus outperform vanilla transformers given the same computational budget [8]. However, the final layer does not operate at the same granularity as the input, making it unusable for tasks that require this such as per-token classification or generative tasks. Hourglass transformers [22] include both up- and down-sampling mechanisms, resulting in a computational saving whilst still being general-purpose models.

3 Methodology

3.1 Latent Dataset Generation

We use the standard two-stage scheme for vector-quantized image modelling [38, 26, 12, 3] using VQ-GAN [12] as our feature extractor. Where such models are available, we use pretrained VQ-GANs for our experiments. For higher resolution experiments (for example, FFHQ-1024 [18]), pretrained models are not available and so training our own VQ-GAN was necessary (see §3.3).

The second stage is to learn a discrete prior model over these latent variables. To enable this, we must first build a latent dataset using our trained VQ-GAN. Formally, given a dataset of images $\mathcal{X}$, a VQ-GAN encoder E with downsample factor f, and vector-quantization codebook $\mathcal{C}$ with number of codewords v, trained on $\mathcal{X}$, we define our latent dataset $\mathcal{L}$ as:

$$\mathcal{L} = \{\mathcal{C}(E(\mathbf{x})) \mid \mathbf{x} \in \mathcal{X}\} \tag{3}$$

where $\mathbf{x} \in \mathbb{R}^{3\times H\times W}$ is a single element of the image dataset and $\mathbf{z} = \mathcal{C}(E(\mathbf{x})) \in \{1, \dots, v\}^{h\times w}$ is the corresponding discrete latent representation. In other words, each $f \times f$ pixels in $\mathbf{x}$ is mapped to a single discrete value from 1 to $|\mathcal{C}|$ (which in turn, corresponds to a vector $\mathbf{e} \in \mathcal{C}$), resulting in a latent representation of shape $\frac{H}{f} \times \frac{W}{f} = h \times w$.

We then use $\mathcal{L}$ to train a discrete prior over the latents. Coupled with the VQ-GAN decoder G, we obtain a powerful generative model.

3.2 2D-Aware Hourglass Transformer

Inspired by successes in hierarchical transformers for generative language modelling [22], we modify their architecture for use with discrete latent representations of image data. We will later use this architecture as the discrete prior over the VQ-GAN latents.

Hourglass transformers have been seen to efficiently handle long-sequences, outperform existing models using the same computational budget, and meet the same performance as existing models more efficiently by using an explicit hierarchical structure [22]. The same benefits should also apply to vector-quantized image modelling.

2D-Aware Downsampling – The original formulation of hourglass transformers [22] introduced both upsampling and downsampling layers, allowing the use of hierarchical transformers in tasks that have output sequence length equal to the input sequence length. However, applying their proposed resampling strategies directly on the vector-quantized image may not be the best strategy. Resampling is applied to flattened token sequence, meaning that the corresponding two-dimensional vector-quantized image is actually resampled more in one axis compared to the other. In their work they did not address this, except for experiments on ImageNet32 [27] where they resampled with a rate of $k = 3$, corresponding to three colour channels.

In our formulation, we instead reshape the flattened sequence back into a two-dimensional form and then apply resampling equally in the last two axes. With a resampling rate of k we apply $\sqrt{k}$ in

each axis. We found this to significantly improve the performance of the discrete prior model, and suspect a similar approach could improve performance if applied to pixels directly, which we leave for future work.

Rotary Positional Embeddings [32] are a good default choice for injecting positional information into transformer models, requiring no additional parameters. Additionally, they can be easily extended to the multi-dimensional case [2] which we do here. Though transformers are clearly capable of learning that elements far apart in a flattened sequence may be close in a multi-dimensional final output, we find that explicitly extending positional embeddings to the multi-dimensional case to provide a modest boost in performance.

Removal of Causal Constraints – In the original autoregressive formulation of hourglass transformers, great care was taken to avoid information leaking during resampling, and hence making the model non-causal [22]. We use a non-autoregressive method which is therefore not causal. Hence, in our approach we do not make any special considerations to avoid information leaking into the future.

3.3 Training a megapixel VQ-GAN

Training at higher resolutions usually means greater computational requirements and sampling speeds. With an autoregressive model, the sampling speed can be especially immense, even with an auxiliary vector-quantized image model [12]. With a non-autoregressive model however, one question to explore is whether sampling at very high resolutions becomes feasible.

To answer this question, we train a larger variant of VQGAN with $v = 8192$ operating on 1024×1024 RGB images. To our knowledge, this is the highest resolution VQGAN has been applied to [12]. Once trained, can generate a latent datasets as before, the only difference being an increased sequence length – greater than was ever tested in the original work [28].

3.4 Non-Autoregressive Generator Training

We train a SUNDAE model on the flattened extracted VQ latents $\mathbf{z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_N\}$ where $N = h \cdot w$. The function f_θ is implemented using our 2D-aware hourglass transformer. Though the default $T = 2$ performed well, we found $T = 3$ to result in more diverse results during unconditional generation.

3.5 Generating High-Resolution Images

During sampling, we simply sample sequentially $\mathbf{z}_t \sim f_\theta(\mathbf{z}_t|\mathbf{z}_{t-1})$ for a constant number of steps T, beginning randomly from $\mathbf{z}_0$ [28]. The original work proposed a number of improved strategies for sampling in smaller number of steps, including low-temperature sampling and updating a random subset of tokens [28], rather than all simultaneously.

Sampling, however, with a lower temperature can reduce the diversity of the resulting samples. To alleviate this, we instead anneal the temperature down from a high value (≈ 1.0) down to a lower value towards the end of the sampling process. We found this retained the fast sampling speed whilst also improving diversity.

In certain latent sampling configurations, updating only a random subset of tokens does improve performance. However, we found that for low step scenarios ($T < 20$) that all tokens must be able to be updated in order to produce meaningful samples before the maximum number of steps is reached. Hence in these cases, we do not follow this strategy.

Additionally, if a individual sample does not change between step $t-1$ and t, we freeze it, preventing any further change. If all samples are frozen, sampling may terminate early, further improving sampling speed with little cost to sample quality. This is significant when performing large-batch sampling.

Once sampling has terminated, the sampled latent code $\mathbf{z}_T$ can be given to the VQGAN decoder G to produce a final sample $\mathbf{y}$.

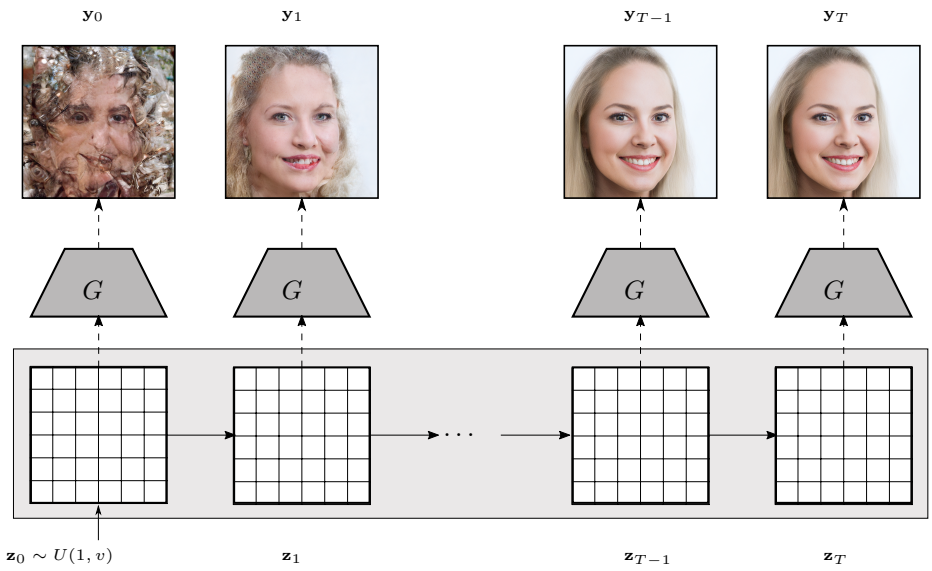

Figure 2: The sampling process. SUNDAE gradually denoises from $\mathbf{z}_0$ to the final sample $\mathbf{z}_T$. At each step, it is possible to decode with G to produce a final image.

Figure 3: An example of inpainting on a 1024×1024 image using our model.

3.6 Arbitrary Pattern Inpainting

As noted in the original work [28] and other non-autoregressive solutions [3] one clear advantage of non-autoregressive models is that they are not limited to causal inpainting. In general, they support arbitrary inpainting masks and can draw on context in both the past and the future, enabling them to perform inpainting tasks not possible with autoregressive sampling.

Given a sampled image $\mathbf{y} \in \mathbb{R}^{H \times W \times 3}$ we can mask a proportion of it using a pixel-level binary mask $m_p \in \{0, 1\}^{H \times W}$. By taking $f \times f$ regions of m_p and applying a logical and in them, we can obtain a latent level mask $m_{\text{vq}} \in \{0, 1\}^{h \times w}$. We then sample as normal from the latents, allowing the model full context, but only update regions that were masked according to m_{vq}. Like with sampling, we then use G to decode the sampled latent code, producing the output $\mathbf{y}$.

4 EVALUATION

4.1 UNCONDITIONAL IMAGE GENERATION

foobar

4.2 ARBITRARY IMAGE INPAINTING

foobar

5 CONCLUSION

In this work, we proposed using denoising autoencoders for the non-autoregressive prediction of VQ latents. This enables fast sampling times and flexible inpainting. In addition, we made changes to the hourglass transformer architecture to make it more suited for two-dimensional signals. Additionally, we demonstrate the scalability of our approach by training a VQ-GAN at extremely high resolutions and training our model on the resulting latent dataset. Ultimately, this allows for the sampling of high quality and diverse 1024×1024 images in mere seconds. Further work is required to improve sampling time further – closing the gap with single-step methods like GANs – and to improve the reconstruction quality of VQ-GAN when operating at high downsampling factors. Additionally, we note that the adversarial component of the VQ-GAN image model may still lead to issues such as mode collapse, which can only be resolved with research into more powerful VQ representation models that do not rely on a adversarial component.

REFERENCES

[1] Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces, 2021. 5

[2] Stella Biderman, Sid Black, Charles Foster, Leo Gao, Eric Hallahan, Horace He, Ben Wang, and Phil Wang. Rotary embeddings: A relative revolution. blog.eleuther.ai/, 2021. [Online; accessed]. 7

[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, 2021. 2, 4, 5, 6, 8

[4] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis, 2018. URL https://arxiv.org/abs/1809.11096. 4

[5] Xi Chen, Nikhil Mishra, Mostafa Rohaninejad, and Pieter Abbeel. Pixelsnail: An improved autoregressive generative model, 2017. 1, 3, 4

[6] Rewon Child. Very deep vaes generalize autoregressive models and can outperform them on images, 2020. URL https://arxiv.org/abs/2011.10650. 4

[7] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches, 2014. 3

[8] Zihang Dai, Guokun Lai, Yiming Yang, and Quoc V. Le. Funnel-transformer: Filtering out sequential redundancy for efficient language processing, 2020. 6

[9] Cyprien de Masson d'Autume, Mihaela Rosca, Jack Rae, and Shakir Mohamed. Training language gans from scratch, 2019. URL https://arxiv.org/abs/1905.09922. 4

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. 5

[11] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis, 2021. 2

[12] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis, 2021. 2, 4, 6, 7

[13] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014. 1, 3

[14] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020. 2

[15] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. 3

[16] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax, 2016. 5

[17] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks, 2018. URL https://arxiv.org/abs/1812.04948. 4

[18] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks, 2019. 6

[19] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013. 1, 4

[20] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization, 2022. 4, 5

[21] Jen-Yu Liu, Yu-Hua Chen, Yin-Cheng Yeh, and Yi-Hsuan Yang. Unconditional audio generation with generative adversarial networks and cycle regularization, 2020. 4

[22] Piotr Nawrot, Szymon Tworkowski, Michał Tyrolski, Łukasz Kaiser, Yuhuai Wu, Christian Szegedy, and Henryk Michalewski. Hierarchical transformers are more efficient language models, 2021. 2, 6, 7

[23] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Łukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer, 2018. 1, 3

[24] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks, 2012. 3

[25] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation, 2021. 1

[26] Ali Razavi, Aaron van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2, 2019. 2, 4, 6

[27] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge, 2015. 6

[28] Nikolay Savinov, Junyoung Chung, Mikolaj Binkowski, Erich Elsen, and Aaron van den Oord. Step-unrolled denoising autoencoders for text generation, 2022. 2, 5, 7, 8

[29] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution, 2019. 2

[30] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations, 2020. 2

[31] Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models, 2021. 2

[32] Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2021. 7

[33] Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder, 2020. URL https://arxiv.org/abs/2007.03898. 4

[34] Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space, 2021. 2

[35] Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks, 2016. 3

[36] Aaron van den Oord, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu. Conditional image generation with pixelcnn decoders, 2016. 3

[37] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning, 2017. 2, 4, 5

[38] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning, 2018. 6

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. 2, 3, 6

[40] Alex Wang and Kyunghyun Cho. Bert has a mouth, and it must speak: Bert as a markov random field language model, 2019. 5

[41] Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion gans, 2021. 1, 2

[42] Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved vqgan, 2021. 4, 5

[43] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. Soundstream: An end-to-end neural audio codec, 2021. 4, 5

[44] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks, 2017. URL https://arxiv.org/abs/1703.10593. 4