Megapixel Image Generation with Step-unrolled Denoising Autoencoders

Alex F. McKinney* Chris G. Willcocks
Department of Computer Science
Durham University, UK

Figure 1. High-resolution samples produced using our non-autoregressive approach. Each 1024×1024 sample was generated in ≈ 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.

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.

1. Introduction

foobar

2. Related Work

This work builds upon much prior research into powerful deep generative models [4], self-supervised methods, and efficient transformer architectures. We briefly cover relevant prior work into deep generative models in §2.1-2.3 and a recent and highly effective development into a efficient transformer architecture in §2.4.

2.1. Autoregressive Generative Models

One major deep generative model family is autoregressive 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. 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 autoregressive model $p_\theta(\cdot)$ can generates new samples sequentially:

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

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 autoregressive models are order-agnostic, allow for arbitrary ordering to be used during training and inference.

2.2. Non-autoregressive Generative Models

Non-autoregressive generative models include generative adversarial networks (GANs), score-based (SBMs) and diffusion models, 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 need to use the chain rule) the actual number of steps varies greatly: from single-step generation in GANs to potentially many thousands in the original diffusion model literature.

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. Step-unrolled Denoising Autoencoder

One recent non-autoregressive model is step-unrolled denoising autoencoders (SUNDAE) [12] 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 [12]. 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 [16] denoising objective 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:

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

[12] 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 [12]. 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 [12]. Typically, $T = 2$ during training, as a single step would be similar to the training strategy of BERT [6] but would lead to worse performance as seen in earlier work using BERT as a random field language model [16].

The training objective of SUNDAE is simply the average of all reconstruction losses $L^{(1:T)}(\theta) = \frac{1}{T}\left(L^{(1)}(\theta) + \dots + L^{(T)}(\theta)\right)$ of the chain after t steps, which is shown to form an upper bound on the actual negative log-likelihood [12]. Taking more steps T leads to a minor improvement in performance, but considerably slows down training time [12] and increases memory usage.

One advantage of this approach is that sampling starts from random tokens, rather than a dedicated "masking" token [1, 3]. 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.

2.4. Hourglass Transformers

Vanilla transformers incur a hefty memory and time complexity of $O(L^2)$ for each block [15]. 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 [5] progressively downsamples the input sequence and hence reduces the computational cost of the model. The saved FLOPs can then be reassigned to create deeper or wider models and thus outperform vanilla transformers given the same computational budget [5]. 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 [9] 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 [3, 7, 10, 14] using VQ-GAN [7] 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 [8]), pretrained models are not available and so training our own VQ-GAN was necessary (see §3.6).

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 $\mathbf{z} = \mathcal{C}(E(\mathbf{x})) \in \{1, \ldots, v\}^{h\times w}$ 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 [9], 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 [9]. The same benefits should also apply to vector-quantized image modelling.

2D-Aware Downsampling – The original formulation of hourglass transformers [9] 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 [11] 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 [13] 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.

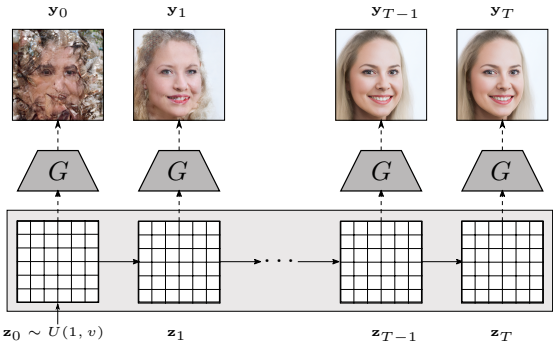

Figure 2. The sampling process. SUNDAE gradually denoises from $\mathbf{z}_0$ to the final sample $\mathbf{z}_T$. At each step, it is possible to decode with G to produce a final image.

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 [9]. 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. Non-Autoregressive Generator Training

3.4. 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 $\mathbf{z}_0$ [12]. 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 [12], 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-

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

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 $\mathbf{y}$.

3.5. Arbitrary Pattern Inpainting

As noted in the original work [12] 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 $\mathbf{y} \in \mathbb{R}^{3\times H\times W}$ 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 $\mathbf{y}$.

3.6. 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 [7]. 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 VQ-GAN with $v = 8192$ operating on 1024×1024 RGB images. To our knowledge, this is the highest resolution VQ-GAN has been applied to [7]. Once trained, we can train SUNDAE samplers on the latents as before, the only difference being an increased sequence length – greater than was ever tested in the original work [12].

4. Evaluation

4.1. Unconditional Image Generation

foobar

4.2. Arbitrary Image Inpainting

foobar

5. Conclusion

FUBAR

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. 2

[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]. 3

[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

[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. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, page 1–1, 2021. 1

[5] Zihang Dai, Guokun Lai, Yiming Yang, and Quoc V. Le. Funnel-transformer: Filtering out sequential redundancy for efficient language processing, 2020. 2

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. 2

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

[8] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks, 2019. 2

[9] Piotr Nawrot, Szymon Tworkowski, Michał Tyrolski, Łukasz Kaiser, Yuhuai Wu, Christian Szegedy, and Henryk Michalewski. Hierarchical transformers are more efficient language models, 2021. 2, 3

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

[11] 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. 3

[12] Nikolay Savinov, Junyoung Chung, Mikolaj Binkowski, Erich Elsen, and Aaron van den Oord. Step-unrolled denoising autoencoders for text generation, 2022. 2, 3, 4

[13] Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2021. 3

[14] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning, 2018. 2

[15] 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

[16] Alex Wang and Kyunghyun Cho. Bert has a mouth, and it must speak: Bert as a markov random field language model, 2019. 2