

StraIT: Non-autoregressive Generation with Stratified Image Transformer

Shengju Qian^{1,3*} Huiwen Chang¹ Yuanzhen Li¹ Zizhao Zhang²
 Jiaya Jia³ Han Zhang¹
¹Google Research ²Google Cloud AI ³CUHK

Abstract

We propose *Stratified Image Transformer (StraIT)*, a pure non-autoregressive (NAR) generative model that demonstrates superiority in high quality image synthesis over existing autoregressive (AR) and diffusion models (DMs). In contrast to the under-exploitation of visual characteristics in existing vision tokenizer, we leverage the hierarchical nature of images to encode visual tokens into stratified levels with emergent properties. Through the proposed image stratification that obtains an interlinked token pair, we alleviate the modeling difficulty and lift the generative power of NAR models. Our experiments demonstrate that StraIT significantly improves NAR generation and out-performs existing DMs and AR methods while being order-of-magnitude faster, achieving FID scores of 3.96 at 256×256 resolution on ImageNet without leveraging any guidance in sampling or auxiliary image classifiers. When equipped with classifier free guidance, our method achieves an FID of 3.36 and IS of 259.3. In addition, we illustrate the decoupled modeling process of StraIT generation, showing its compelling properties on applications including domain transfer.

1. Introduction

Image generation has recently achieved significant progress, demonstrating prominence in content creation, editing and many other applications. With increasing data and computational resources, leading methods, such as *diffusion models* [7, 20, 36, 40, 44] and *autoregressive transformers* [9, 41, 56, 57], have largely surpassed prior works based on generative adversarial networks (GANs) [1, 12, 26, 58] in both image quality and diversity. For example, despite being in different model families, diffusion models like ADM [7] and autoregressive models like VIM [56] all beat GANs in class-conditional generation. Similarly, DALL-E [40, 41], Parti [57] and Imagen [44] have shown unprecedented photorealism compared to GANs for text-to-image synthesis.

*This work was done during an internship at Google Research. Correspondence to: Han Zhang <zhanghan@google.com>, Shengju Qian <sjqian@cse.cuhk.edu.hk>. Code and models will be released.

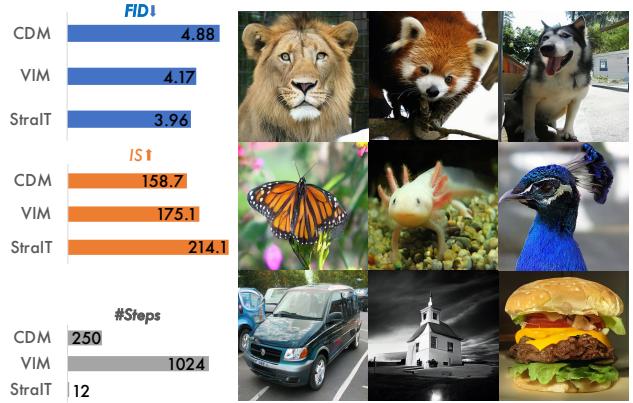


Figure 1. Quantitative comparison with leading AR and DMs on ImageNet 256² generation: VIM [56] and CDM [21], as well as conditional generated samples from StraIT at the resolution of 512². The methods are compared consistently *without leveraging classifiers, rejection sampling, or classifier free guidance*. More visualizations can be found at Appendix.

Albeit with impressive power, most *autoregressive* (AR) and *diffusion models* (DMs) are compute-demanding and have slow sampling speeds, a bottleneck that impedes their accessibility in practical applications. Specifically, DMs commonly require hundreds or thousands of successive evaluation steps in inference, to gradually reduce the noise. AR transformers, on the other hand, need to sequentially decode an image following the raster scan ordering, *i.e.* from left to right and line-by-line. These steps are not parallelizable, resulting in high inference latency. Though several works have explored different strategies [32, 34, 45, 47] to reduce the sampling steps, they usually sacrifice image quality for faster speed.

Non-autoregressive transformers [2, 60], where tokens are decoded in parallel, have contemporarily been explored and demonstrated both promising generation quality and efficiency. For example, MaskGIT [2] leads to significant inference acceleration while achieving competitive sample quality when trained with the mask-then-predict objective [6] and equipped with iterative decoding [11]. Despite considerable progress, however, leading *non-autoregressive* models still lag behind state-of-the-art *diffusion* [21] and *autoregres-*

sive [28, 56] counterparts in public benchmarks.

On the other hand, Vector Quantization(VQ) [9, 52], which reduces computational complexity through spatial compression, has been widely used in modeling high resolution images. However, a large downsampling rate trades reconstruction quality for efficiency, setting an unavoidable bottleneck for VQ-based generation. LDM [43] recently suggests a relatively mild compression rate for DMs, but it results in much larger latent spatial resolution and longer sequence for modeling in the following stage. The surge in computational costs and memory requirements for handling longer sequences prevent its adoption in NAR transformers. Moreover, recent works in NLP [13, 16] also show that long sequence modeling is one of the central challenges of non-autoregressive generation.

In this paper, we propose *StraIT*, a stratified non-autoregressive model motivated by the actual human painting process. With our tokenizer adaptation, distinctive but interlinked top-level and bottom-level token sequences are obtained from images. Most importantly, this improved token hierarchy intriguingly presents a *short-but-complex* top and *long-but-simple* bottom arrangement reflected in perplexity, relieving the difficulty on modeling longer sequences. With the proposed *Cross-scale Masked Token Modeling* strategy, both the top and bottom-level modules are trained with masked visual token prediction, whereas the second one models the top-to-bottom conditional probability.

We make the following three main contributions:

- We demonstrate the difficulty on scaling up non-autoregressive model from larger models and longer sequences. To improve NAR generation, we exploit visual characteristics and investigate a suitable tokenization strategy through image stratification.
- With the interlinked token pairs, we propose a stratified modeling framework named *StraIT*, and empirically demonstrate that, for the first time, our pure NAR method significantly out-performs existing state-of-the-art AR and DMs in on the ImageNet benchmark while achieving $30\times$ faster inference.
- Furthermore, we conduct extensive ablation studies and provide insights into the generation process of *StraIT*, demonstrating emergent properties of the decoupled procedure, where the top and bottom transformers own notably different responsibility. We demonstrate that these compelling properties also provide the versatility of *StraIT* to perform semantic domain transfer, with simple forward passes.

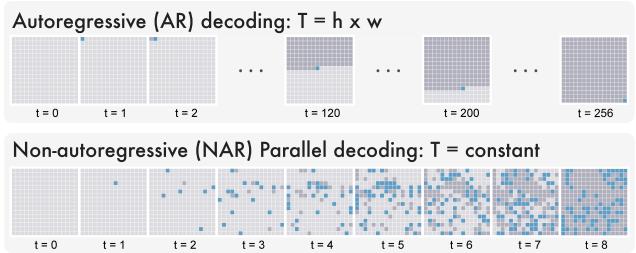


Figure 2. Autoregressive vs Non-autoregressive Decoding.

2. Background

2.1. Non-autoregressive Image Generation

It is computationally infeasible to directly model pixel dependencies for high-resolution images. Most of recent *non-autoregressive* [2, 60] image generative models adopt the two-stage approach, which consists of a visual tokenization stage and a masked modeling stage.

Visual Tokenization In this stage, the goal is to compress the image into discrete and spatially-reduced latent space. The model consists of three major parts: an encoder E , a quantizer Q with a learnable codebook e and a decoder G . Given an RGB image I with spatial resolution (H, W) , the encoder E first extracts visual features with resolution $(H/f, W/f)$, where f is the downsampling ratio. Then the quantizer Q performs a nearest neighbor look-up in the codebook e to quantize latent features into discrete codes. Then the decoder G takes the corresponding features of discrete codes and maps them back to the pixel space to reconstruct the original image. All these three modules are trained together with reconstruction [52] or adversarial [9] objectives. After training, the encoder can extract discrete latent codes for second-stage generative modeling.

Masked Token Modeling with Transformers Similar with the masked language modeling (MLM) task introduced in BERT [6], the objective in masked token modeling is also used to predict masked image tokens. Instead of using a fixed masking ratio in language and visual pre-training [6, 17], the generative transformer needs to generate tokens from scratch and applies a randomly sampled ratio $\gamma(r) \in (0, 1]$ in training. Given a sampled binary mask $\mathbf{m} \in \{m_1, \dots, m_K\}^K$, the token y_i is replaced with [MASK] if $m_i = 1$, and remains intact when $m_i = 0$. Let $Y_{\overline{\mathbf{M}}}$ denote the masked token sequences. Conditional input \mathbf{c} , such as class label, is concatenated as a prefix to $Y_{\overline{\mathbf{M}}}$. The objective of training a NAR transformer parameterized by θ is to minimize the negative log-likelihood of the masked positions:

$$\mathcal{L}_{\text{mask}}(\theta) = - \mathbb{E}_{\mathbf{Y} \in \mathcal{D}} \left[\sum_{\forall i \in [1, K], m_i=1} \log p(y_i | \mathbf{c}, Y_{\overline{\mathbf{M}}}) \right]. \quad (1)$$

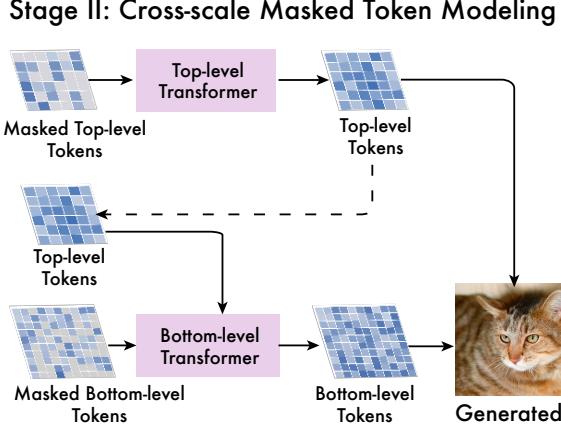
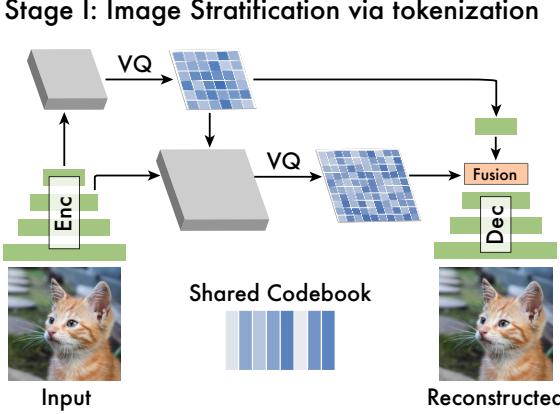


Figure 3. **Pipeline Overview.** In the first stage, we obtain stratified visual token sequences using our proposed *VQGAN2-R*. With the *short-but-complex* top and *long-but-simple* bottom token pairs produced by our tokenizer, we design *Cross-scale Masked Token Modeling* to train our top and bottom-level transformers in a decoupled manner. The dotted line denotes a teacher forcing regime in bottom training.

Iterative Refinement During Inference NAR models can in theory infer all tokens in a single pass, but the performance lags largely behind autoregressive generation. Parallel decoding algorithm [2, 11, 28, 29], as illustrated in Figure 2, has been studied in both NLP and vision domains, which offers improved fidelity and diversity while maintains much faster inference than autoregressive counterparts [9, 56]. Since our focus is different, we simply adopt the confidence-based strategy in MaskGIT [2] and propose our stratified iterative decoding in Section 3.3.

2.2. Problems of NAR Image Generation

Despite much faster sampling speed, leading NAR methods [2, 29] still under-perform state-of-the-art AR and DMs [7, 28, 56] in sample quality. Moreover, AR and DMs have achieved much progress by scaling up the architectures [21, 44, 57] or applying mild downsampling rates to latent spaces [43, 56]. However, we find that directly scaling to larger models (Section 4.2) and longer sequences (Section 4.3) for NAR is not sufficient to unravel this performance gap.

3. Stratified Image Transformer

In this section, we provide a novel framework called StraIT to improve NAR model for high quality image synthesis. In Section 3.1, we study the tokenization step and carefully design *image stratification*, the key technique that enables fine-grained control. In Section 3.2, we elaborate our strategy of *decoupled non-autoregressive modeling*.

3.1. Image Stratification via Tokenization

Generative transformers majorly process vision contents in a language-modeling style, where an image [8, 41, 56], video [10, 23], or other structured inputs [3, 33] are processed into a sequence of discrete vocabulary tokens. How-

ever, these contents are usually formed with different hierarchies, *e.g.* from subpixels to edges, which are unfortunately neglected in this style. In this work, we leverage image hierarchy for sequence modeling to achieve better generation results. Namely, we decompose an image into two stratified representations, which inherently reduces the difficulty on modeling long sequences.

Given an image $I \in \mathbb{R}^{H \times W \times 3}$, our tokenizer outputs two spatial collections of codebook entries, \mathbf{Y}_{top} and \mathbf{Y}_{bottom} , where $\mathbf{Y}_{top} \in \mathbb{R}^{\frac{H}{16} \times \frac{W}{16}}$ and $\mathbf{Y}_{bottom} \in \mathbb{R}^{\frac{H}{8} \times \frac{W}{8}}$. They are then flattened to 1D, resulting in a short top sequence $Y_t = [y_t^i]_{i=1}^N$ and a long bottom sequence $Y_b = [y_b^i]_{i=1}^{4N}$.

Stratified Tokenization As depicted in Figure 3, we utilize a two-level token hierarchy. For 256×256 images, the encoder E first transforms and downsamples the inputs by factors of 8 and 16, obtaining feature representations of 32×32 and 16×16 respectively. The 16×16 latent map is firstly quantized to our top-level codes. The latent conditional layer, which consists of several residual blocks, upscales the quantized top-level map and then stack it with the 32×32 features. After the final bottom-level quantization, we obtain Y_t and Y_b , two stratified token sequences with different lengths.

For the encoder structure, we follow the design choice in VQVAE-2 [42]. While for the decoder applied with different fusion strategies, we observe distinctive representations.

To differentiate, we adopt two variants: *VQGAN2-C*, which follows the strategy in VQVAE-2 [42], fusing top and bottom levels together by concatenation; and *VQGAN2-R*, which processes the bottom level as residuals with *stratified residual fusion*, *i.e.* firstly upsample the top-level features, and then add the bottom-level features onto it.

Training Objectives. Following VQGAN [9], we apply the perceptual loss [25] and adversarial loss [9] in seeking perceptual quality. For simplicity, we use the shared codebook to quantize top and bottom features, where the commitment loss [52] is applied to both layers. Thus, the total vector quantization training loss is,

$$\mathcal{L}_{\text{VQ}}(E, G, \mathbf{e}) = \mathcal{L}_{\text{Adv}} + \mathcal{L}_{\text{Perc}} + \mathcal{L}_{\text{Commit}}(Y_b, Y_t). \quad (2)$$

Model	#Tokens	FID/PSNR	Perplexity (PPL)	
			Top	Bottom
VQGAN ($f=16$) [9]	16^2	2.04/19.9	5893 ± 2.1	
VQGAN ($f=8$) [9]	32^2	0.81/24.3	6387 ± 2.8	
VQGAN2-C	$16^2 + 32^2$	0.65/24.9	405 ± 1.3	6428 ± 4.1
VQGAN2-R	$16^2 + 32^2$	0.67/24.8	5632 ± 2.0	1644 ± 1.9

Table 1. Comparison between *VQGAN2-C* and *VQGAN2-R*, as well as our re-implemented VQGAN ($f=8, 16$) with a consistent recipe. All four models have a codebook size of 8192.

Fusion Strategy and Emergent Properties. We provide quantitative evaluation of our stratified tokenizer in Table 1, and qualitative comparison in Appendix B. Compared to single-scale tokenization, *VQGAN2-C* and *VQGAN2-R* perform slightly better by introducing extra tokens. To understand the distinction between the tokens learned by them, we report the per-batch perplexity (PPL) during training. Serving as one of the most common metrics for evaluating language models, VQ perplexity measures the codebook utilization [50] and reflects the complexity of sequences:

- *VQGAN2-C*: the model exhibits a ‘greedy’ property, where the bottom level has a high PPL and top level has an extremely low PPL, showing it is rarely exploited.
- *VQGAN2-R*: conversely, the bottom level works as a residual to the top level, making the top level has a much higher PPL than bottom.

While previous works such as VQVAE-2 [42] adopted hierarchical codes for more powerful priors over the latent codes, our focus is to obtain a suitable conditional distribution for non-autoregressive modeling on longer sequences. In other words, we want to relieve the burdens on modeling longer sequences non-autoregressively. Accordingly, we adopt *VQGAN2-R* as it provides interlinked *short-but-complex* top and *long-but-simple* bottom token representations. We also provide detailed comparisons in Section 4.3.

3.2. Cross-scale Masked Token Modeling

With stratified tokenizer E and de-tokenizer G available, we can now represent an image with two dependent sequences. In contrast to previous sequence modeling framework, we propose to learn two decoupled transformers by *Cross-scale Masked Token Modeling* (CMTM). In the following, we illustrate the process, training objectives, and inference techniques.

Decoupled Modeling Let $Y^t = [y_i^t]_{i=1}^N$ denote the obtained top-level tokens, where N is the length of the flatten matrix, and $Y^b = [y_i^b]_{i=1}^{4N}$ represents the respective bottom-level tokens. Instead of separate or joint modeling, we choose to model them in a decoupled manner:

- *Top-level transformer*: serves as a likelihood model that tries to generate top-level tokens purely from scratch.
- *Bottom-level transformer*: models the conditional likelihood, and learns to predict the corresponding bottom-level tokens given top-level inputs.

Training Objectives We adopt *Cross-scale Masked Token Modeling* to train the top and bottom-level model parameterized by θ^t and θ^b . For the top-level transformer, the task is to predict masked tokens directly. While for the bottom-level transformer, strong conditional guidance is provided by top-level tokens, which makes the modeling process easier. Let \mathbf{m}^t and \mathbf{m}^b denote the independently sampled masks. The top-level and bottom-level masked token modeling losses are:

$$\mathcal{L}_{\text{mask}}^{\text{top}}(\theta^t) = - \mathbb{E}_{\mathbf{Y}^t \in \mathcal{D}} \left[\sum_{\substack{m_i^t=1, \\ \forall i \in [1, N]}} \log p(y_i^t | \mathbf{c}, Y_{\bar{M}}^t) \right], \quad (3)$$

$$\mathcal{L}_{\text{mask}}^{\text{bot}}(\theta^b) = - \mathbb{E}_{\mathbf{Y}^b \in \mathcal{D}} \left[\sum_{\substack{m_i^b=1, \\ \forall i \in [1, 4N]}} \log p(y_i^b | \mathbf{c}, Y^t, Y_{\bar{M}}^b) \right]. \quad (4)$$

In practice, these two objectives enable separate training of the top and bottom-level transformers, reducing much memory cost. Such simple yet effective stratified modeling doesn’t require conditional augmentations [21], which we elaborate more in Appendix C.

3.3. Inference with StraIT

Stratified Iterative Decoding We follow the iterative parallel decoding in CMLM [11] and MaskGIT [2] to generate images. In contrast to previous non-autoregressive methods, we generate two stratified sequences in a top-down manner. Specifically, the top-level transformer predicts all tokens starting from a blank canvas \bar{M} where all tokens are masked out. Each refinement step fills the canvas with a number of tokens according to their predicted probability. The completely predicted top-level sequence then guides the bottom-level transformer to perform its conditional iterative decoding on \bar{N} following a similar procedure.

Our decoding process is illustrated as follows:

Require: $\bar{M}, \bar{N} = \emptyset, T^{\text{top}}, T^{\text{bottom}}, \gamma$

- 1: **for** $t \leftarrow 1$ to T^{top} **do**
- 2: $n = \gamma(t, T^{\text{top}}), \hat{y}_i^t \sim P_{\theta^t}(y_i^t | \widehat{\mathcal{Y}}_{\bar{M}}^t, \mathbf{c}), \forall i \in M$
- 3: $\bar{M} \leftarrow \bar{M} \cup \{\arg \operatorname{topk}_{i \in M}(P_{\theta^t}(y_i^t | \widehat{\mathcal{Y}}_{\bar{M}}^t, \mathbf{c}), k=n)\}$
- 4: **end for**
- 5: **for** $t \leftarrow 1$ to T^{bottom} **do**

```

6:    $n = \gamma(t, T^{\text{bottom}})$ ,  $\hat{y}_i^b \sim P_{\theta^b}(y_i^b | \widehat{\mathcal{Y}}_{\overline{N}}, \overline{M}, \mathbf{c})$ ,  $\forall i \in N$ 
7:    $\overline{N} \leftarrow \overline{N} \cup \{\arg \max_{i \in N} (P_{\theta^b}(y_i^b | \widehat{\mathcal{Y}}_{\overline{N}}, \overline{M}, \mathbf{c}), k=n)\}$ 
8: end for

```

where T^{top} , T^{bottom} denote the steps of top and bottom-level decoding. For γ , we adopt a cosine function from [2]. We study the allocation of decoding steps in Section 4.3.

4. Experiments

In this section, we evaluate the performance of Strait on image generation, in terms of quality-diversity, efficiency, and adaptability. In Section 4.2, we provide both quantitative and qualitative evaluations on the standard class-conditional image generation on ImageNet [5]. In Section 4.3, we conduct ablation studies to understand our stratified modeling process, as well as showing its advantages over different variants. Then in Section 4.4, we analyze the intriguing property of our system, and show its compelling applications.

4.1. Experimental Setup

For all experiments, we train our tokenizer *VQGAN2-R* with a single codebook of 8192 tokens using cropped 256×256 images from ImageNet [5]. The images are always downsampled by factors of 16 and 8, respectively producing top and bottom-level tokens. The codebook is also used to train our model on 512×512 .

For *Cross-scale Masked Token Modeling*, we leverage two different transformer architectures, including the top-level transformer that consists of self-attention blocks and the bottom-level transformer that adopts the cross-attention blocks. Sharing embedding dimensions of 768, the models used in this work have the following configuration:

- Top: 48 layers, an intermediate size of 5120, and 32 attention heads, leading to 499M parameters.
- Bottom: 16 layers, an intermediate size of 4096, and 24 attention heads, leading to 292M parameters.

We train baselines and two models proposed using AdamW [31] with a base learning rate of 1e-4, adopting a 5000-step linear warmup and a cosine decaying schedule afterward. Label smoothing [49] and dropout [48] are also employed following [2]. Each model is trained for 200 epochs with a batch size of 256 on TPU chips. To allow fair comparisons and investigate influence from larger models, we adopt our consistent recipes to train MaskGIT [2] with 1.3B parameters, which receives marginal improvement shown in Table 2 and indicates the inefficiency to naively scaling model sizes in existing NAR paradigms.

4.2. Main Results on Image Synthesis

Table 2 and Table 3 summarize the main results of Strait for 256×256 class-conditional generation on ImageNet [5]. Since improved inference strategies have been explored for

different generative models, it's difficult to compare them in a unified way. Therefore, we conduct comparisons without any guidance in training and inference in Table 2 and then incorporate classifier-free guidance [22] to Strait, reporting the results in Table 3.

Quantitative Evaluation From Table 2, we show that, without any special sampling methods, our method significantly out-performs previous state-of-the-arts in both Fréchet Inception Distance (FID) [18] and Inception Score (IS). For the first time, we demonstrate that non-autoregressive model out-performs advanced autoregressive [27, 56] and diffusion [21] models with much fewer steps. In addition, our method maintains a trade-off between precious and recall, which suggests a better coverage (Recall) compared to MaskGIT [2], and improved sample quality (Precision) to diffusion models.

While our focus is not to introduce improved sampling methods, we incorporate the commonly used classifier free guidance [22] by DMs to our method. Specifically, we adopt a probability of randomly dropping conditioning in training as 0.1, and a guidance scale of 0.2 during inference. As demonstrated in Table 3, our method achieves the best FID and IS on ImageNet generation on record when not using classifiers or rejection sampling, further showing its strength.

Higher Resolution in Token-level To demonstrate the versatility of non-autoregressive model, we follow recent cascade diffusion models [7, 21, 43] to generate higher resolution *i.e.* 512×512 in a purely non-autoregressive manner. For simplicity, we utilize our same tokenizer trained on 256×256 . Using the same architecture of bottom-level transformer, we train another upsampling model to generate the stratified tokens representing 512×512 images conditioned on the lower resolution tokens obtained by bilinear downsampled 256×256 images. Due to the memory cost, we adopt a simple three-layer model denoted by U . During inference, the generated sequences from Strait is fed into U for iterative decoding to tokens for 512×512 outputs.

We provide results on ImageNet 512×512 conditional generation in Table 4. Our simple three-layer model performs significantly better than ADM- U [7] that adopts a more intensive upsampling strategy. Notably, it also outperforms ADM- U -G in terms of IS, which adopts pre-trained classifier guidance. Note that this comparison does not intend to push performance on 512×512 , rather suggesting the prominence of pure NAR generation.

User Preference Study To showcase the capabilities of NAR formalism, we follow CDM [21] to compare with other methods directly, *i.e.* avoid using external image classifiers to boost sample quality. However, recent works like StyleGAN-XL [46] that adopted classifier guidance achieve

Family	Method	# Params	# Steps	FID ↓	IS ↑	Precision ↑	Recall ↑
Autoregressive	VQVAE-2 [42]	13.5B	5120	31.11	45	0.36	0.57
	VQGAN [9]	1.3B	256	15.78	78.3	-	-
	RQ-Transformer [27]	3.8B	256	7.55	137	-	-
	VIT-VQ+VIM [56]	1.7B	1024	4.17	175.1	-	-
Diffusion Models	Improved DDPM [36]	280M	250	12.26	-	0.70	0.62
	ADM [7]	554M	250	10.94	101.0	0.69	0.63
	VQ-Diffusion [15]	518M	100	11.89	-	-	-
	LDM [43]	400M	250	10.56	103.49	0.71	0.62
	CDM [21]	~1B	250	4.88	158.71	-	-
Non-autoregressive	MaskGIT [2]	227M	8	6.18	182.1	0.80	0.51
	MaskGIT [†]	1.3B	12	5.84	180.3	0.73	0.54
	MaskGIT [†]	1.3B	36	5.71	185.9	0.73	0.56
	StraIT	863M	12*	3.96	214.1	0.74	0.62

Table 2. Quantitative comparison with state-of-the-art generative models on ImageNet 256×256 **without leveraging any guidance or external classifiers for training and inference**. For VQ-based methods, ‘# Params’ includes the parameters of VQ. ‘# Steps’ denotes the number of forward runs to generate a sample. [†] denotes the re-implementations with the same setup with ours. * our method adopts an (18+6)-step allocation, which is faster than a 12-step inference of our whole model. Details are provided in Sec. 4.3 and Table 7.

Method	Params	Steps	FID ↓	IS ↑
w. Learnable guidance				
Improved VQ-Diffusion [51]	510M	100	4.83	-
Token Critic [29]	391M	18×2	4.69	174.5
DPC [30]	391M	180	4.45	244.8
w. Classifier-free guidance				
f-DM [14]	302M	250	6.8	-
Draft-and-Revise [28]	1.4B	$68 \times (1+4)^{\dagger}$	3.41	224.6
StraIT	863M	(18+6) [†]	3.36	259.3

Table 3. Comparison on methods with improved inference strategies, where StraIT adopts a classifier-free guidance [22] scale of 0.2. [†][28] adopts a light depth transformer for the 4 steps in (1+4).

Method	T_{base}	T_{upsample}	FID ↓	IS ↑
Token Critic [29]	36	-	6.80	182.1
ADM [7]	250	-	23.24	58.06
ADM-U	250	250	9.96	121.78
ADM-U-G	250	250	3.85	221.72
StraIT-U	12	12	3.82	253.6

Table 4. Results of 512×512 image generation on ImageNet. T_{base} and T_{upsample} denote the number of step to perform lower-scale generation and upsampling, respectively.

promising quantitative results. To allow more systematic and reliable comparisons, we conduct user preference studies using Amazon Mechanical Turk on verifying the quality and diversity of generated samples from four leading methods on ImageNet 512×512 generation: StyleGAN-XL [46] and ADM-U-G [7] that adopt classifier guidance, as well as MaskGIT [2] and Token Critic [29].

For *quality* evaluation which selects the more realistic one, each grader is presented with two randomly sampled images of a same class side-by-side, one using StraIT, the other using one of the competing methods. For *diversity*

evaluation, two batches of twelve generated images from same classes are provided in the same way, where the graders choose the more diverse-looking one. For both quality and diversity, each pairwise comparison is rated by 15 graders.

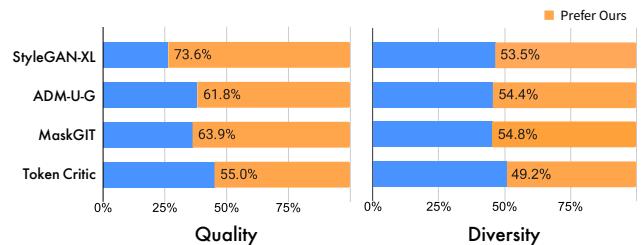


Figure 4. **User preference study** on quality and diversity. The number denotes the portion of times when StraIT is preferred over competing methods.

As shown in Table 4, our method is more preferred than all other competing methods in terms of *quality*. Comparing to these existing best-performing models on ImageNet generation ranging from GANs, transformers, and diffusion models, our method shows a clear advantage despite the simple structure. For *diversity*, it appears that existing methods are capable of generating diverse samples, which are relatively harder for graders to distinguish. However, our method still shows favoured *diversity* over state-of-the-art GAN and DMs that exploit pre-trained classifier: StyleGAN-XL [46] and ADM-U-G [7], as well as leading generative transformers [2, 29].

4.3. Ablation Studies

To better understand our stratified process, we conduct ablation studies and provide detailed illustration. For consistency, we don’t use classifier-free guidance in this section.

Scaling up NAR on Sequence Lengths As investigated in LDM [43], aggressive spatial compression, *i.e.* $f=16$ in VQGAN [9], significantly eliminates high frequency in images. Therefore, applying relatively mild downsampling rates would notably improve quality in diffusion model. By replacing the original $f = 16$ VQGAN with $f = 8$ in MaskGIT [2] and with increased training and inference costs, we show in Table 5 that the final performance is conversely decreased. We conjecture that the conditional independence assumption of NAR models becomes more problematic when dealing with longer sequences. This result also serves as the motivation of our stratified modeling.

# Tokens	FLOPs	Training Costs	FID ↓	IS ↑
16 × 16	48.2	1×	6.05	203
32 × 32	216.5	2.6×	6.61	177.7

Table 5. Results of MaskGIT [2] model trained with different sequence lengths. Training costs are measured on same TPUs.

VQGAN2-R over VQGAN2-C While they share identical architectures apart from the fusion strategy, the proposed VQGAN2-R behaves distinctively with VQGAN2-C. As discussed in Section 3.1 and Table 1, different fusion strategy leads to distinctive token representations. Most importantly, we want to reduce modeling complexity on longer sequences, therefore adopting the *long-but-simple* visual tokens produced by VQGAN2-R. To tell the difference, we provide the results by replacing our tokenizer with VQGAN2-C and keep other architectures consistent. In Table 6, we show that VQGAN2-C leads to much worse FID scores, suggesting that such simple modification deteriorates the transformer significantly. This gap also indicates the importance of proper tokenizers [9] for generative vision transformer, which has been rarely studied.

Stratified over Cascade Another option to improve longer sequences modeling is to use a cascade pipeline that generates sequences progressively, similarly as the upsampling strategy [7, 21] in DMs. To consistently compare in 256×256 resolution, we utilize two tokenizers of VQGAN($f=8, 16$) from Table 1 to construct a cascade variant. Using a similar pipeline in our CMTM, we replace the top and bottom-level tokens with 16×16 ones from VQGAN($f=16$) and 32×32 ones from VQGAN($f=8$). As this top-level code can generate images using its own decoder, we report the results on ImageNet 256×256 in Table 6 from both the whole system and top-level transformer. We find that the modified cascade pipeline helps improve the results of $f=16$ baseline marginally, expressing the validity of adopting guidance for longer sequences from shorter ones. In contrast to this coarse-to-fine design, our stratified tokenizer extracts hierarchical and interlinked visual token sequences, which significantly benefit *Cross-scale Masked Token Modeling*.

Type	Tokenizer	Transformers		FID ↓	Δ
		Top	Bottom		
Stratified	<i>VQGAN2-C</i>	✓	✓	6.11	-
	<i>VQGAN2-R</i>	✓	✓	3.96	-2.15
Cascade	VQGAN ($f=16$)	✓		6.05	
	VQGAN ($f=8$)	✓	✓	5.71	-0.34

Table 6. Results of 256×256 image generation on ImageNet from different tokenizers and pipelines. The results from Cascade $f=16$ are obtained by decoding predicted tokens from Top-level.

Decoding Steps Unlike previous generative transformers [2, 6, 38], our paradigm results in two decoupled sequences. Following the iterative decoding strategy [11] and cosine mask scheduling [2], we study the effect from different allocations on decoding steps in Table 7. Interestingly, spending more decoding steps on the top level benefits both FID and IS. Considering the teacher forcing training regime on the bottom level, it makes sense to ‘pay attention to the condition’. It’s note-worthy that the top model, despite with more parameters, operates on short 16×16 sequences and requires less computation than the bottom-level architecture which operates on 32×32 sequences *i.e.* top: 133.5G FLOPS, bottom: 311.3G FLOPS. As it does not reflect actual inference speed, we estimate the wall-clock time over 50000 generation samples. Given better results and notable speedup, we adopt 18+6 decoding steps for our experiments.

T^{top}	T^{bottom}	FID ↓	IS ↑	Speedup
3	21	5.4	193	0.6×
6	18	5.6	192	0.7×
9	15	4.5	201.3	0.9×
12	12	4.21	202.7	1×
15	9	4.05	214	1.2×
18	6	3.97	214.1	1.5×
21	3	4.0	209	1.6×

Table 7. Comparisons on decoding steps. T^{top} and T^{bottom} represent the top-level and bottom-level decoding steps. Speedup is estimated by generating 50000 images on TPUv3.

4.4. Intriguing Properties and Applications

In this section, we first illustrate the stratified decoding process of StraIT, showing an intriguing decoupled property. Then we show the versatility of StraIT on doamin transfer, without any architecture changes or fine-tuning.

Stratified Modeling Process To better understand our stratified modeling process, we further provide visualizations of the inference steps. As shown in Figure 5, the top-level transformer gradually infills basic colors, coarse boundaries, and general layout of an image, while the bottom-level model refines detailed textures to these regions and produces realistic images. This intriguing phenomenon empirically explains the stratified information encoded in our top-level

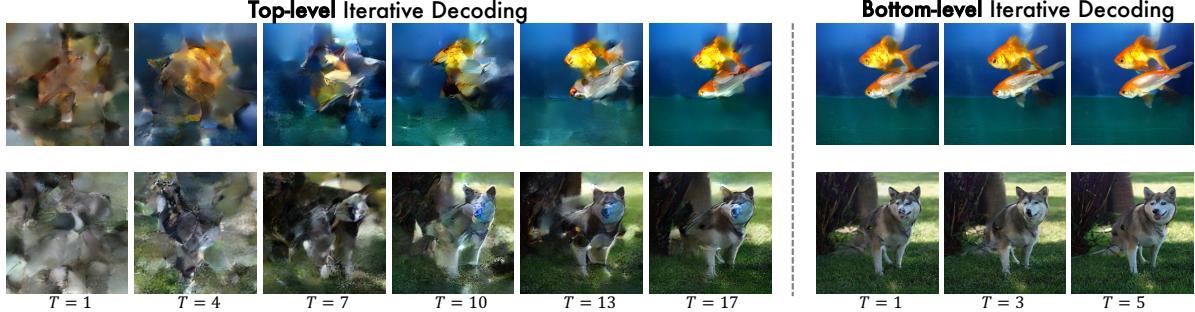


Figure 5. **Intermediate outputs at different steps of our stratified Iterative Decoding.** T on the left and right denotes the current step of top and bottom-level decoding. For visualizing the top-level iterative decoding process while the bottom-level code is not available, we provide a fixed randomly-sampled bottom-level tokens to the decoder.

and bottom-level visual tokens. Moreover, the decoupled generation process is analogous to human painting, where an image draft is firstly constructed by rough brushstrokes, and then the bottom-level transformer, serving like an expert editor, gradually embroiders the entire image.

Non-autoregressive models trained with *Masked Token Modeling* have a nature of learning to infill. Therefore, they can be seamlessly applied to multiple image editing tasks by handling them as constraints to the input mask. Previous works on NAR model [2] have demonstrated its versatility of image inpainting and extrapolation. Recently, Phenaki [53] also adopts NAR for variable length video generation. Our work serves as a stronger NAR framework and also share similar advantages. Instead of echoing these benefits, we show emerging applications from our framework.

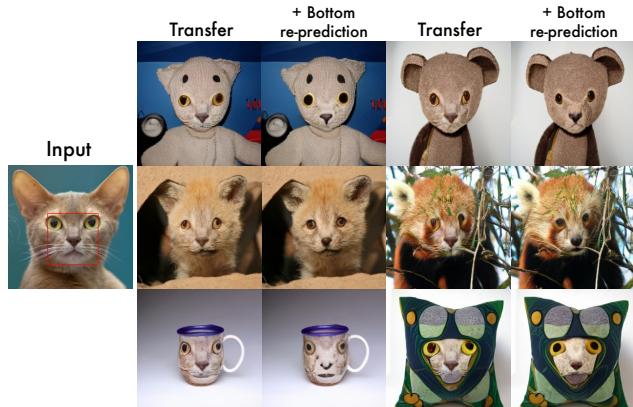


Figure 6. **Transferring a cat to different semantic domains.** Targeted domains include: [Teddy] \times 2, [Kit Fox], [Red Panda], [Coffee Mug], [Pillow]. **Zoom in for details.**

Semantic Domain Transfer One compelling benefit of StraIT is to perform domain transfer. Since StraIT enables decoupled modeling which provides better semantic abstraction, we can now perform domain transfer easily by masking unwanted tokens. As shown in Figure 6, our method successfully transfers the chosen geometry and skeleton to diverse

domains, even with large semantic variance.

Bottom-level Re-prediction As validated in Figure 5, the bottom-level transformer is designed to perform detailed refinement given the top-level visual tokens, akin to the original texture. To allow transfer adaptability, one simple strategy is to mask all bottom-level tokens and leverage the transformer to perform re-prediction from the transferred top-level tokens. With no extra computation incurred, such simple strategy benefits domain transfer clearly and naturally, from the comparison in Figure 6. In supplement to Figure 5, the results further elaborate our stratified modeling: the top level performs semantic understanding and generate coarse layout; while the bottom level edits and re-touches on the visual details.

These intriguing properties of StraIT open up many possibility of applying NAR to generation and editing, especially considering its fast decoding and simple formalism.

5. Related Works

Visual Tokenization By converting an image into a sequence of discrete codes, Vector Quantization [41, 52] has been adopted to obtain visual tokens. Recent works such as VQGAN [9] and ViT-VQGAN [56] try to maintain fine details with improved techniques, while RQ-VAE [27, 28], on the other hand, represents the image as a stacked map of discrete codes. Prior works have also investigated hierarchical VQ for learning powerful priors [37, 42], representing multi-level textures for human synthesis [24], or to attain high factors of compression [54].

Non-autoregressive Generation While recent generative models, including AR and DMs, show improved results on both images [40, 44, 57] and videos [19], another family of non-autoregressive models [2, 13, 53, 60] with accelerated inference have also been explored. Different from recent works [29, 30] that investigated better sampling strategies, our work targets at proposing a new NAR paradigm that

handles longer sequences. Recently, [28] incorporated AR with NAR generation, showing another direction of hybrid NAR modeling.

6. Conclusion

In this work, we propose StraIT, a stratified non-autoregressive generative paradigm that out-performs existing autoregressive and diffusion model on class-conditional image generation. The proposed strategies of image stratification and *Cross-scale Masked Token Modeling* allow predicting longer visual tokens sequences that include more fine-grained details. The promising results, order-of-magnitude faster inference, and versatility to applications make StraIT an attractive choice for generative modeling. Our studies indicate that the non-autoregressive family is a promising direction for future research.

Acknowledgements

We would like to thank José Lezama for discussions and generous suggestions, as well as Ruben Villegas for the helpful reviews and constructive feedback on the draft.

References

- [1] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.
- [2] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *CVPR*, 2022.
- [3] Ting Chen, Saurabh Saxena, Lala Li, David J Fleet, and Geoffrey Hinton. Pix2seq: A language modeling framework for object detection. *arXiv preprint arXiv:2109.10852*, 2021.
- [4] Giannis Daras and Alexandros G Dimakis. Discovering the hidden vocabulary of dalle-2. *arXiv preprint arXiv:2206.00169*, 2022.
- [5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.
- [6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [7] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *NeurIPS*, 2021.
- [8] Ming Ding, Zhuoyi Yang, Wenyi Hong, Wendi Zheng, Chang Zhou, Da Yin, Junyang Lin, Xu Zou, Zhou Shao, Hongxia Yang, et al. Cogview: Mastering text-to-image generation via transformers. *NeurIPS*, 2021.
- [9] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, 2021.
- [10] Songwei Ge, Thomas Hayes, Harry Yang, Xi Yin, Guan Pang, David Jacobs, Jia-Bin Huang, and Devi Parikh. Long video generation with time-agnostic vqgan and time-sensitive transformer. *arXiv preprint arXiv:2204.03638*, 2022.
- [11] Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. Mask-predict: Parallel decoding of conditional masked language models. *EMNLP*, 2019.
- [12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014.
- [13] Jiatao Gu, James Bradbury, Caiming Xiong, Victor OK Li, and Richard Socher. Non-autoregressive neural machine translation. *arXiv preprint arXiv:1711.02281*, 2017.
- [14] Jiatao Gu, Shuangfei Zhai, Yizhe Zhang, Miguel Angel Bautista, and Josh Susskind. f-dm: A multi-stage diffusion model via progressive signal transformation. *arXiv preprint arXiv:2210.04955*, 2022.
- [15] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In *CVPR*, 2022.
- [16] Junliang Guo, Xu Tan, Di He, Tao Qin, Linli Xu, and Tie-Yan Liu. Non-autoregressive neural machine translation with enhanced decoder input. In *AAAI*, 2019.
- [17] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022.
- [18] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *NIPS*, 2017.
- [19] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022.
- [20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 2020.
- [21] 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.
- [22] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.
- [23] Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. *arXiv preprint arXiv:2205.15868*, 2022.
- [24] Yuming Jiang, Shuai Yang, Haonan Qiu, Wayne Wu, Chen Change Loy, and Ziwei Liu. Text2human: Text-driven controllable human image generation. *TOG*, 2022.
- [25] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016.
- [26] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020.
- [27] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *CVPR*, 2022.

- [28] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Draft-and-revise: Effective image generation with contextual rq-transformer. *arXiv preprint arXiv:2206.04452*, 2022.
- [29] José Lezama, Huiwen Chang, Lu Jiang, and Irfan Essa. Improved masked image generation with token-critic. In *ECCV*, 2022.
- [30] Jose Lezama, Tim Salimans, Lu Jiang, Huiwen Chang, Jonathan Ho, and Irfan Essa. Discrete predictor-corrector diffusion models for image synthesis. In *ICLR*, 2023.
- [31] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- [32] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *arXiv preprint arXiv:2206.00927*, 2022.
- [33] Jiasen Lu, Christopher Clark, Rowan Zellers, Roozbeh Mottaghi, and Aniruddha Kembhavi. Unified-io: A unified model for vision, language, and multi-modal tasks. *arXiv preprint arXiv:2206.08916*, 2022.
- [34] Chenlin Meng, Ruiqi Gao, Diederik P Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. *arXiv preprint arXiv:2210.03142*, 2022.
- [35] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *ICML*, 2018.
- [36] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *ICML*, 2021.
- [37] Shengju Qian, Yi Zhu, Wenbo Li, Mu Li, and Jiaya Jia. What makes for good tokenizers in vision transformer? *T-PAMI*, 2022.
- [38] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.
- [39] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.
- [40] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.
- [41] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *ICML*, 2021.
- [42] Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *NeurIPS*, 2019.
- [43] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.
- [44] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. *NeurIPS*, 2022.
- [45] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.
- [46] Axel Sauer, Katja Schwarz, and Andreas Geiger. Stylegan-xl: Scaling stylegan to large diverse datasets. In *SIGGRAPH* 2022, 2022.
- [47] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.
- [48] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 2014.
- [49] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016.
- [50] Yuhta Takida, Takashi Shibuya, WeiHsiang Liao, Chieh-Hsin Lai, Junki Ohmura, Toshimitsu Uesaka, Naoki Murata, Shusuke Takahashi, Toshiyuki Kumakura, and Yuki Mitsu-fuji. Sq-vae: Variational bayes on discrete representation with self-annealed stochastic quantization. *arXiv preprint arXiv:2205.07547*, 2022.
- [51] Zhicong Tang, Shuyang Gu, Jianmin Bao, Dong Chen, and Fang Wen. Improved vector quantized diffusion models. *arXiv preprint arXiv:2205.16007*, 2022.
- [52] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *NIPS*, 2017.
- [53] Ruben Villegas, Mohammad Babaeizadeh, Pieter-Jan Kindermans, Hernan Moraldo, Han Zhang, Mohammad Taghi Saffar, Santiago Castro, Julius Kunze, and Dumitru Erhan. Phenaki: Variable length video generation from open domain textual description. *arXiv preprint arXiv:2210.02399*, 2022.
- [54] Will Williams, Sam Ringer, Tom Ash, David MacLeod, Jamie Dougherty, and John Hughes. Hierarchical quantized autoencoders. *NeurIPS*, 2020.
- [55] Yuxin Wu and Kaiming He. Group normalization. In *ECCV*, 2018.
- [56] 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. *arXiv preprint arXiv:2110.04627*, 2021.
- [57] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunnar Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, et al. Scaling autoregressive models for content-rich text-to-image generation. *arXiv preprint arXiv:2206.10789*, 2022.
- [58] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *ICML*, 2019.
- [59] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *ICCV*, 2017.
- [60] Zhu Zhang, Jianxin Ma, Chang Zhou, Rui Men, Zhikang Li, Ming Ding, Jie Tang, Jingren Zhou, and Hongxia Yang. M6-ufc: Unifying multi-modal controls for conditional image synthesis via non-autoregressive generative transformers. *arXiv preprint arXiv:2105.14211*, 2021.

The content of the appendix is organized as follows:

- Qualitative results from StraIT in Appendix. A
- Discussion on Stratified Tokenization in Appendix. B
- Discussion on Decoupled Transformer in Appendix. C
- More experimental details in Appendix. D
- Limitations and future works in Appendix. E

A. Qualitative Results from StraIT

In this section, we provide more qualitative results to show the effectiveness of StraIT.

Image Generation We provide 512×512 image generation results of different classes from ImageNet in Figure 8 and 9. In order to demonstrate the sample quality and diversity, we provide multiple samples given one class. With much smaller number of inference steps, StraIT provides diverse samples with high visual quality.

Flexible Image Editing Since NAR models trained with masked token modeling have the nature of infilling missing contents, StraIT is capably of editing image flexibly in simple feedforward passes by tokenizing unmasked tokens. We provide diverse image editing results in Figure 10, including infilling missing regions and replacing with other context.

B. Discussion on Stratified Tokenization

Image stratification, which represents an image with two distinctive sequences of visual tokens, is an important design in our work. We investigate modified tokenizations in order to obtain stratified visual codes *i.e.* the top-level codes decide coarse layout and color distribution with thick strokes, while the bottom level stores refined texture. Such stratified nature further enables *Cross-scale Masked Token Modeling* and leads to the decoupled generation process of StraIT.

Perplexity To compare variants of tokenizers, we adopt the perplexity (PPL) to measure the complexity of visual tokens. While the perplexity metric has been widely used in measuring language models, it denotes codebook utilizations of tokenizers as well as the complexity of token representations in vision. The perplexity is computed as:

$$\text{Perplexity}(p) = 2^{-\sum_x p(x) \log_2 p(x)} = \prod_x p(x)^{-p(x)} \quad (5)$$

Note that we report the per-batch PPL during training, which converges after 700k iterations. The stabilized per-batch PPL indicates the ordinary complexity of visual tokens.



Figure 7. Qualitative comparisons among different tokenizers.

Comparison between VQGAN2-R/C As mentioned, we propose image stratification to provide suitable token sequences for non-autoregressive modeling. The experiment results in Table 5 show that, with longer sequences, NAR transformer actually performs worse than using short ones. While naively increasing sequence length shows inferior performance, we choose to leverage the hierarchical nature in images by representing them into interlinked token pairs. With the guidance from top-level codes, we expect our bottom-level transformer to perform better on modeling longer sequences. However, when training the tokenizer, different fusion strategies in decoder lead to distinctive token representations, as have been quantified in Table 1.

- *VQGAN2-C*, on the one hand, concatenates the bottom level ($f=8$) and the upscaled top level ($f=16$) together, acting greedily to exploit bottom-level code which provides more information with less spatial compression. Similarly as VQVAE2 [42], this bottom-level emphasis leads to poor exploitation in top-level codes, which has also been observed in recent work [24].
- *VQGAN2-R*, in contrast, decodes directly from top level and treats the bottom level as residual by adding it back to the second level of de-tokenizer. Such simple-yet-effective *stratified residual fusion* strategy leads to stratified visual tokens, as the model processes top-level visual codes as stem and relies less on the bottom level.

The differences in perplexity and the visualization of our generation process have reflected the difference between top and bottom-level visual codes. To provide more context, we also conduct qualitative comparison in Figure 7. It's noteworthy that our target is not to build VQ variants with stronger capability or better reconstructions. As shown, both VQGAN ($f=8$) and VQGAN2 perform better than $f=16$, and VQGAN2-R performs slightly worse than VQGAN2-C. However, VQGAN2-R provides more suitable visual tokens for decoupled non-autoregressive modeling, where the *short-but-complex* top and *long-but-simple* bottom arrangement gives much better generation performance.

Figure 8. Selected 512×512 generated samples from StraIT.

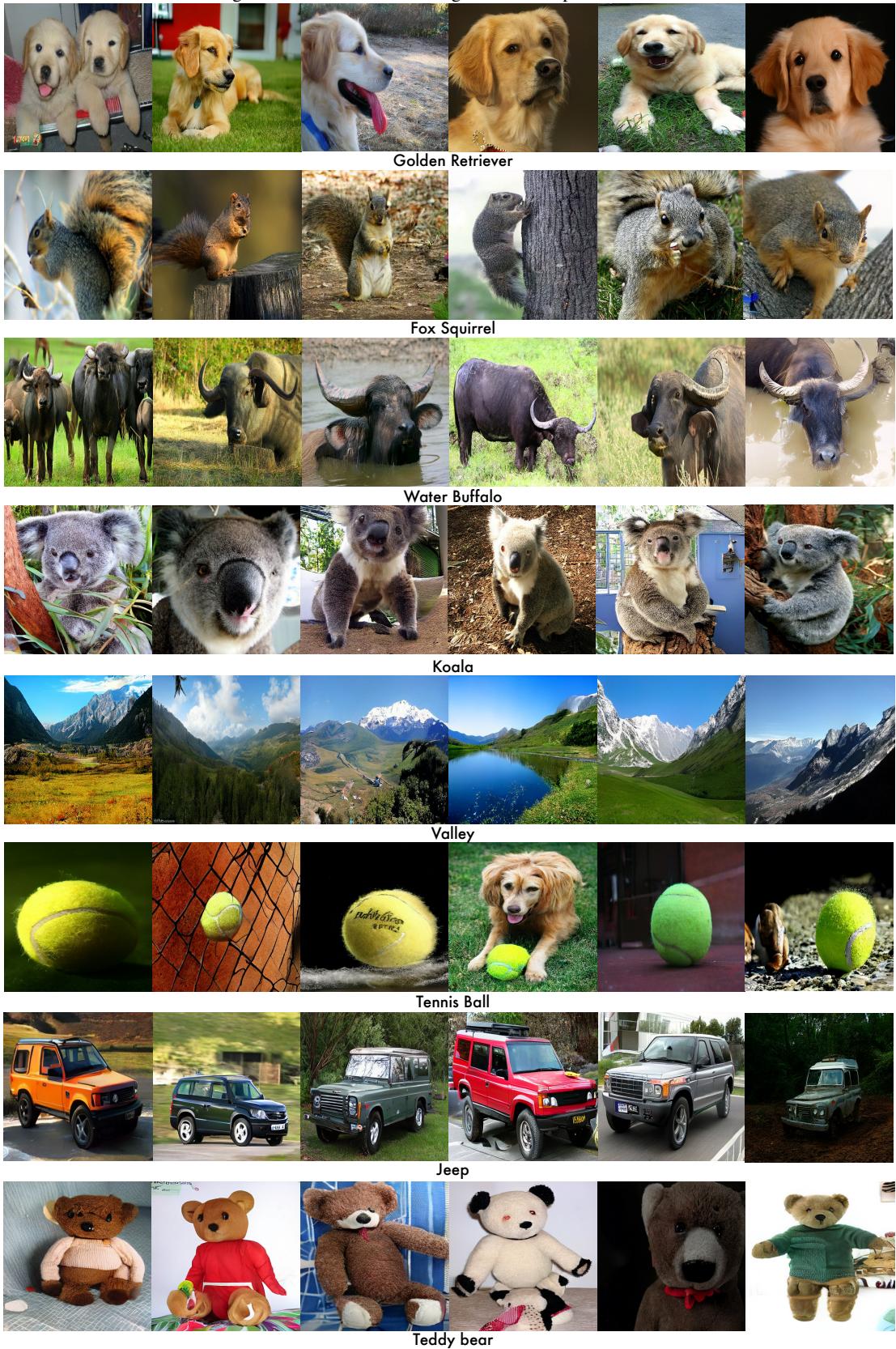
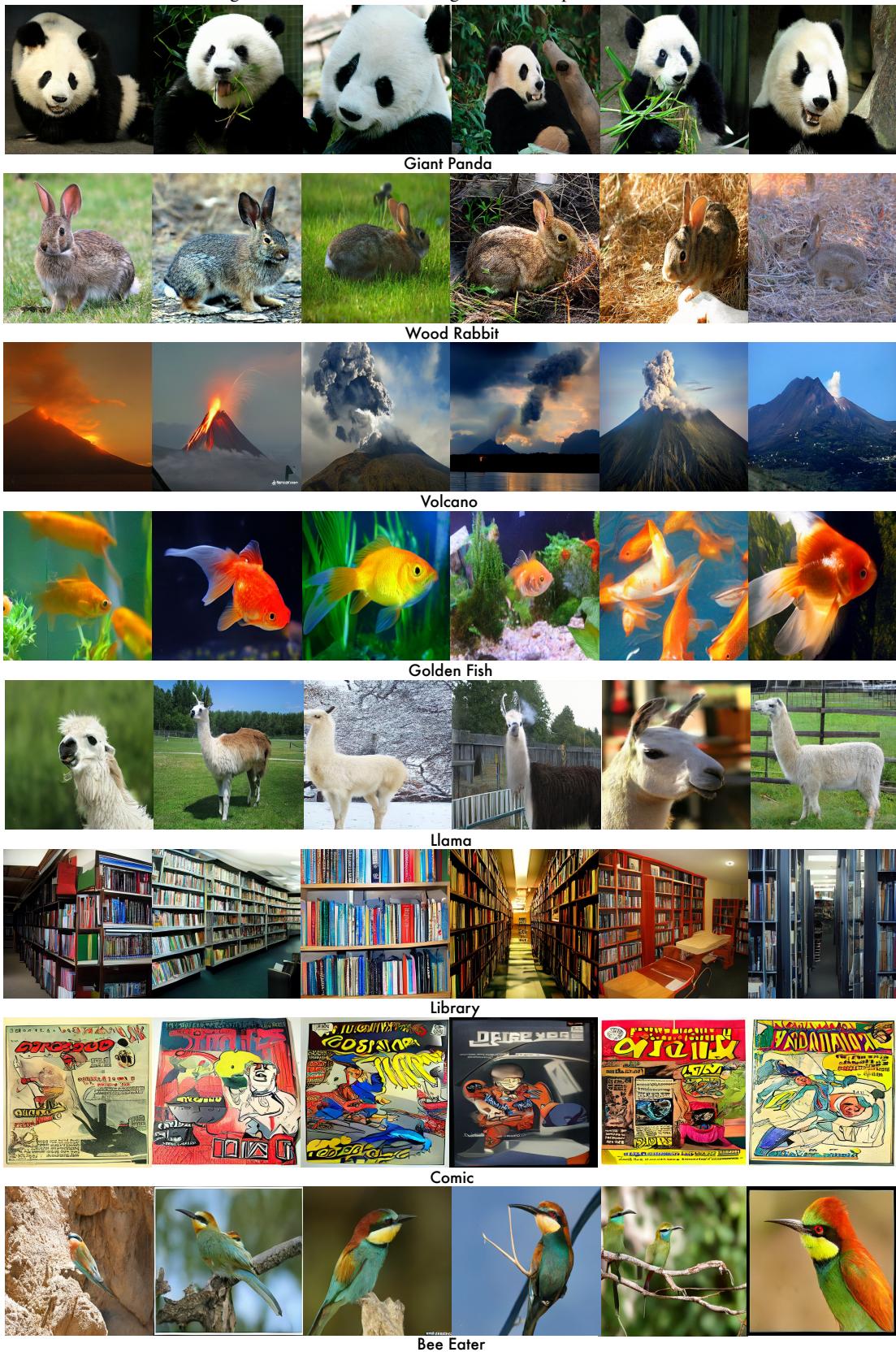


Figure 9. Selected 512×512 generated samples from Strait.



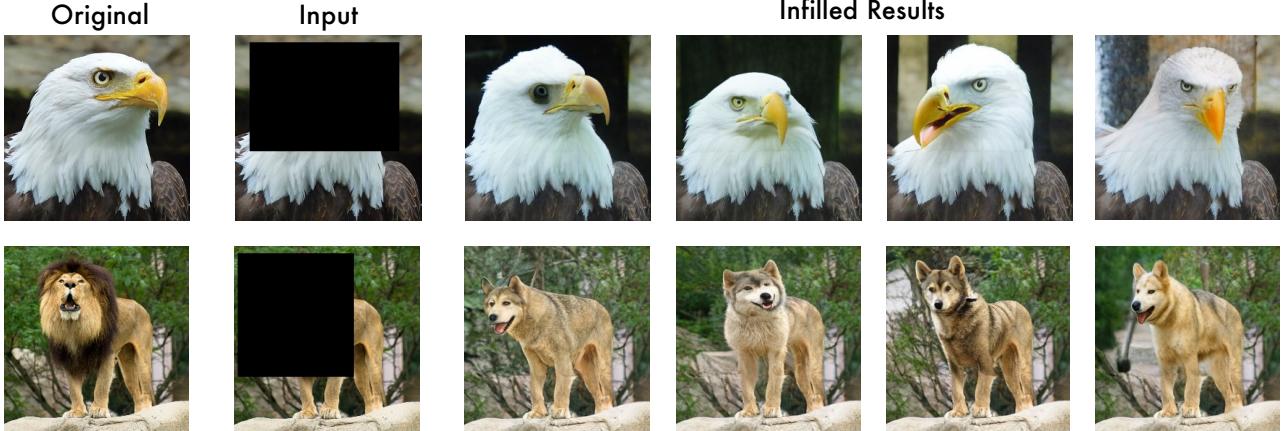


Figure 10. Flexible image infilling result. The second image in each row denotes the input image to our model. The first row represents inpainting results with the same given label. The second row shows the infilled results from different context *i.e.* turning lion into husky.

Method	Codebook Size \mathcal{Z}		FID \downarrow	IS \uparrow
	$\ \mathcal{Z}_{\text{top}}\ $	$\ \mathcal{Z}_{\text{bottom}}\ $		
Shared	8192		3.96	214.1
Separate	8192	2048	3.89	216.3

Table 8. Results on ImageNet 256×256 generation with shared and separate codebook designs, without classifier free guidance.

Separate Codebooks Since our experiments have shown the difference between top and bottom tokens, a possible option is to tokenize each level with a separate codebook that represents dedicated features. Considering the perplexity of *VQGAN2-R*, we replace the shared $\|\mathcal{Z}\| = 8192$ codebook with two separate codebooks of $\|\mathcal{Z}_{\text{top}}\| = 8192$ and $\|\mathcal{Z}_{\text{bottom}}\| = 2048$. Following the consistent experimental settings, we report the performance with separate codebooks in Table 8. The results show that, though with increased number of tokens, using separate codebook only improves the quantitative results slightly. For simplicity, we adopt the shared codebook for top and bottom level in our experiments.

C. Discussion on Decoupled Transformer

As have been discussed, the transformers in StraIT model the stratified visual tokens in a decoupled manner using *Cross-scale Masked Token Modeling*. We’ve also illustrated the respective roles of top and bottom-level transformer in visualizations. Beyond the adaptations we’ve proposed in the main paper, we provide additional investigation on this modeling process.

Conditional Augmentation When training the bottom-level transformer, we adopt a teacher forcing training regime *i.e.* the ground truth top-level code Y^t is used directly as the training condition. However, during inference, the top-level condition is generated from our top-level model,

where domain gaps might exist. Such gaps in conditions have been proven to significantly harm the generation results in GANs [59] and cascaded diffusion model [21, 44]. To reduce this gap, we further study conditional augmentation in training our decoupled architectures.

Different from the practice in cascaded diffusion [21], where the low resolution images are degraded with random permutations to reduce potential exposure bias, the generated top-level sequences in StraIT are semantic-aware. To make the bottom-level model more robust towards varying top-level code during inference, we experiment to randomly mask a subset of the ground truth top-level code, and then exploit the top-level model θ^t to predict the contents on partially-masked regions. Let \widetilde{Y}^t denote the augmented conditions, the objective function becomes:

$$\mathcal{L}_{\text{mask}}^{\text{bot}}(\theta^b) = - \mathbb{E}_{\mathbf{Y}^b \in \mathcal{D}} \left[\sum_{\substack{m_i^b=1, \\ \forall i \in [1, 4N]}} \log p(y_i^b | \mathbf{c}, \widetilde{Y}^t, Y_{\overline{M}}^b) \right]. \quad (6)$$

We study the effect from different strengths *i.e.* masking ratios of conditional augmentations in Table 9. In contrast to the conditional augmentations in diffusion models [21, 44] that play a key role, sample quality and diversity of StraIT doesn’t receive benefits from them. As NAR and DMs appear to behave differently, we choose to not include such augmentations in our framework. The straight-forward teacher forcing regime shows state-of-the-art results.

Mask Scheduling Functions One key design in NAR generation is iterative decoding. Note that we do not focus on finding better inference algorithms in this work, thus following the choice in MaskGIT [2] to use a cosine function in determining mask scheduling *i.e.* the fraction of tokens decoded each iteration. To better understand this process, we provide ablation studies on different functions in Table 10.

Method	Masking ratio	FID ↓	IS ↑
Original	-	3.96	214.1
Augmented	10%	4.03	212.8
	30%	4.05	208.7
	50%	3.99	213.5

Table 9. Results from different masking ratios adopted in conditional augmentation.

Function	γ	FID ↓	IS ↑
Convex	Logarithmic	11.32	167.3
	Square Root	7.89	187.5
-	Linear	5.13	200.3
Concave	Cosine	3.96	214.1
	Square	4.13	209.5
	Cubic	4.97	203.1
	Exponential	4.09	212.7

Table 10. Ablation studies on different mask scheduling functions. Each model uses a (18+6)-step. We report the best FID and IS.

As shown, concave functions generally produce better results than convex ones, suggesting the necessity on using proper decoding techniques.

D. Additional Experimental Details

In this section, we further provide more implementation details about architectures, hyper-parameters, and user preference study.

Implementation Details We provide the detailed model architectures and hyper-parameters of *VQGAN2-R* in Table 11. This recipe is consistently leveraged in training other VQGAN variants in our experiments. For our top and bottom-level transformers, we share the same training settings, as shown in Table 12. Both the tokenizer and transformers are trained on 32 TPU chips. For all model variants, we report their best FID and IS by sweeping the sampling temperatures [2].

User Preference Study We present more details about our user preference study. In Figure 11, we provide the interface of quality evaluation, where the users are presented with two generated images of the same class. Given the provided text prompts, the users are asked to select the one with better quality. The interface of diversity evaluation is provided in Figure 12. Different from evaluating quality, users are shown with 16 random samples from two classes to determine the more diverse groups.

Hyper-parameters	<i>VQGAN2-R</i>
training epochs	200
batch Size	256
optimizer	SGD
learning rate	1e-4
lr schedule	constant
gradient penalty	R1 reg [35]
penalty cost	10.0
commitment cost	0.25
GAN loss weight	0.1
perceptual loss weight	0.1
codebook size	8192
embedding dim [56]	32
activation	Swish [39]
normalization	Group Norm [55]
#channels	128
#res blocks	2
channel multi.	[1, 1, 2, 2, 4]
discriminator	StyleGAN d [26]
norm in d	Group Norm [55]
#channel multi. of d	[1]
blur resample	✓

Table 11. Model architectures and hyper-parameters of *VQGAN2-R*. The training process is performed on 32 TPUs.

Hyper-parameters	Transformers
training epochs	200
batch Size	256
optimizer	AdamW [31]
learning rate	1e-4
weight decay	0.045
momentum	$\beta_1, \beta_2 = 0.9, 0.96$
gradient clip	3.0
label smoothing [49]	0.1
warmup steps	5000
uncond cutoff [22]	0.1

Table 12. Hyper-parameters of the decoupled transformers in Strait. The top and bottom-level model share the same training recipes.

Inference Efficiency As the NAR formalism requires much fewer decoding steps than existing AR and DMs, we also study the actual inference speeds to show the efficiency of Strait. We follow the (18+6)-step allocation in the main paper and conduct inference on TPUs. Note that in order to consistently compare with previous NAR methods [2] in terms of parameters, we regard a single step as going through our whole model: including the top and bottom-level transformer. Therefore, this step arrangement is even

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----

Here are two *images generated from the same input category* using different methods.

Please use the radio buttons above to choose the more **realistic** looking one.



Left looks more realistic



Right looks more realistic

Note that you **must select one** in each tab before you submit.

It would be great if you could check both sets of images carefully, as they may contain subtle artifacts that are not immediately obvious. Thank you so much!

Keyboard shortcuts:

Select Left Image	Shift+left
Select Right Image	Shift+right
Previous Tab	Shift+Tab
Next Tab	Tab

Figure 11. User Interface of quality evaluation.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
---	---	---	---	---	---	---	---	----------	----	----	----	----	----	----	----	----	----

Here are two set of *images generated from the same input categories* using different methods.

Please use the radio buttons above to choose the image set with more **diverse** content, lighting and background.



Top is more diverse



Bottom is more diverse

Note that you **must select one** in each tab before you submit.

It would be great if you could check both sets of images carefully, as they may contain subtle differences that are not immediately obvious. Thank you so much!

Figure 12. User Interface of diversity evaluation.

faster than our marked 12-step inference. However, when comparing with relevant works such as Draft-and-revise [28] or Token-critic [29], StraIT can be clarified as using 24 inference steps. On Google TPUv3 and Nvidia V100, it respectively takes 0.18s and 0.39s for StraIT to sample one image, which is order-of-magnitude faster than existing methods such as ADM [7] and LDM [43].

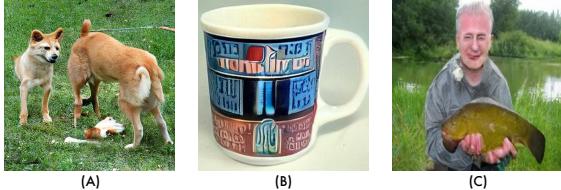


Figure 13. Three generated samples that show limitations, including (A) decoding error; (B) spelling; (C) artifacts on faces and hands.

E. Limitations and future works

Limitations There also exist several limitations in StraIT. In Figure 13, we show some major limitations of our approach. Existing iterative decoding strategies in NAR still have spaces to improve. In (A), we demonstrate a failure case from parallel confidence-based decoding. Predicted tokens with high confidence are kept without replacement, leading to compounding decoding error from early steps. In (B) and (C), we show the model’s incapability on spelling and generating tiny faces or hands, which results in undesired artifacts over these complex structures. These issues are also demanding in existing generative models [4, 43, 57]. Meanwhile, the quadratic computation and memory costs in self attention makes scaling to higher resolution challenging.

Future Works With the improved generation quality and clear inference speedups over AR and DMs, StraIT boosts high fidelity NAR generation and opens up many possibility on practical usages. Serving as a general framework, we expect StraIT to facilitate text-to-image [40, 43, 44, 57] and video generation [53]. More importantly, less inference steps enable optimization and deployment much easier.

Besides these promising directions, the circumstances in limitations also remain future works, including finding better decoding algorithms [29], reducing memory costs, and applying cascaded strategies [21] to enhance details.