Visual Autoregressive Modeling

https://arxiv.org/pdf/2404.02905

Visual Autoregressive Modeling: Scalable Image Generation via Next-Scale Prediction

Keyu Tian[1,2], **Yi Jiang**[2,†], **Zehuan Yuan**[2,*], **Bingyue Peng**[2], **Liwei Wang**[1,*]
[1]Peking University [2]Bytedance Inc
keyutian@stu.pku.edu.cn, jiangyi.enjoy@bytedance.com,
yuanzehuan@bytedance.com, bingyue.peng@bytedance.com, wanglw@pku.edu.cn

Try and explore our online demo at: https://var.vision

Codes and models: https://github.com/FoundationVision/VAR

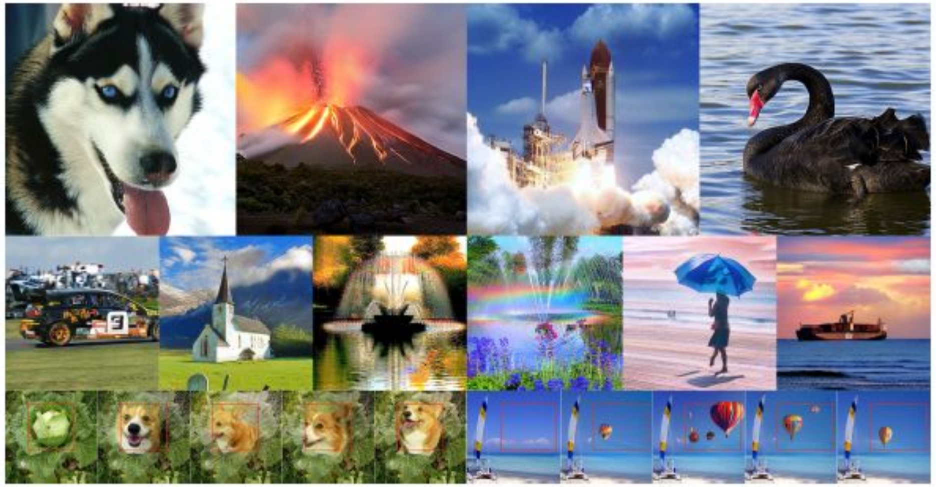


Figure 1: **Generated samples from Visual AutoRegressive (VAR) transformers trained on ImageNet.** We show 512×512 samples (top), 256×256 samples (middle), and zero-shot image editing results (bottom).

(Review: Vision transformer)

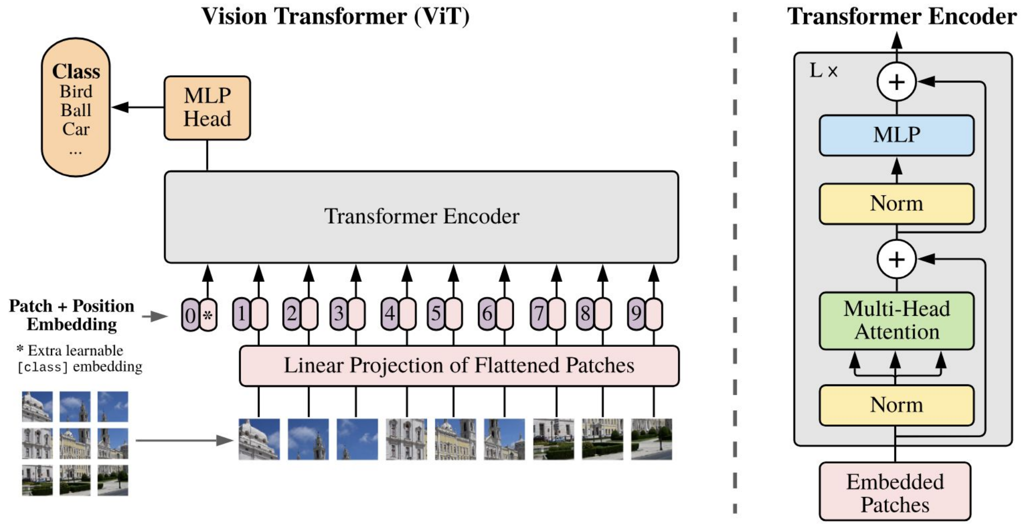
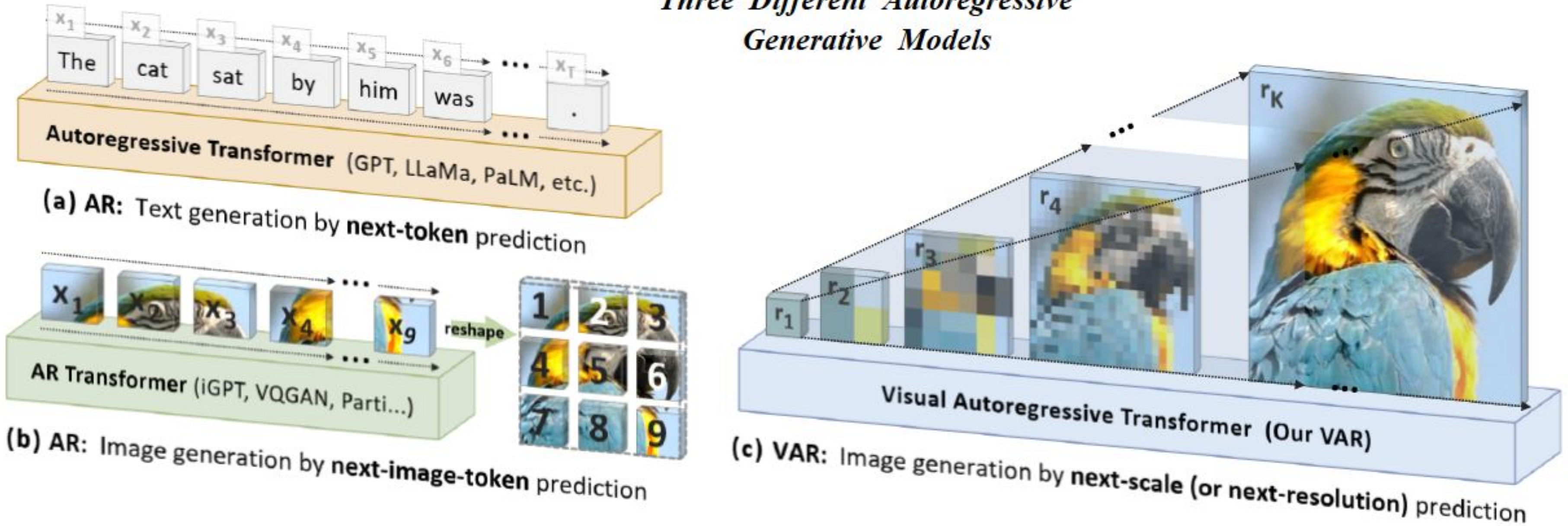


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification token" to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

Three Different Autoregressive Generative Models


x_1 x_2 x_3 x_4 x_5 x_6 ... x_T
The cat sat by him was .
Autoregressive Transformer (GPT, LLaMa, PaLM, etc.)
(a) AR: Text generation by next-token prediction
x_1 x_2 x_3 x_4 ... x_9
reshape
1 2 3 4 5 6 7 8 9
AR Transformer (iGPT, VQGAN, Parti...)
(b) AR: Image generation by next-image-token prediction
r_1 r_2 r_3 r_4 ... r_K
Visual Autoregressive Transformer (Our VAR)
(c) VAR: Image generation by next-scale (or next-resolution) prediction

Autoregressive modeling (I): autoencoder

1. Tokenization with learnable codebook *Z*

$$\text{feature map } f = \text{encoder } \mathcal{E}(\text{image } im)$$

$$q = \text{quantizer } \mathcal{Q}(f) \in [\text{vocab } V]$$

$$q^{(i,j)} = \left(\arg\min_{v \in [V]} \| \text{lookup}(Z, v) - f^{(i,j)} \|_2 \right) \in [V]$$

2. Flatten *q* into 1D sequence

$$x = (x_1, \ldots, x_{h \times w})$$

3. Reconstruction by decoder *D*

$$\hat{f} = \text{lookup}(Z, q), \qquad \hat{im} = \mathcal{D}(\hat{f}), \tag{4}$$

$$\mathcal{L} = \|im - \hat{im}\|_2 + \|f - \hat{f}\|_2 + \lambda_{\text{P}}\mathcal{L}_{\text{P}}(\hat{im}) + \lambda_{\text{G}}\mathcal{L}_{\text{G}}(\hat{im}), \tag{5}$$

where $\mathcal{L}_{\text{P}}(\cdot)$ is a perceptual loss such as LPIPS [97], $\mathcal{L}_{\text{G}}(\cdot)$ a discriminative loss like StyleGAN's discriminator loss [46], and λ_{P}, λ_{G} are loss weights. Once the autoencoder $\{\mathcal{E}, \mathcal{Q}, \mathcal{D}\}$ is fully trained, it will be used to tokenize images for subsequent training of a unidirectional autoregressive model.

Autoregressive modeling (II): next-token prediction

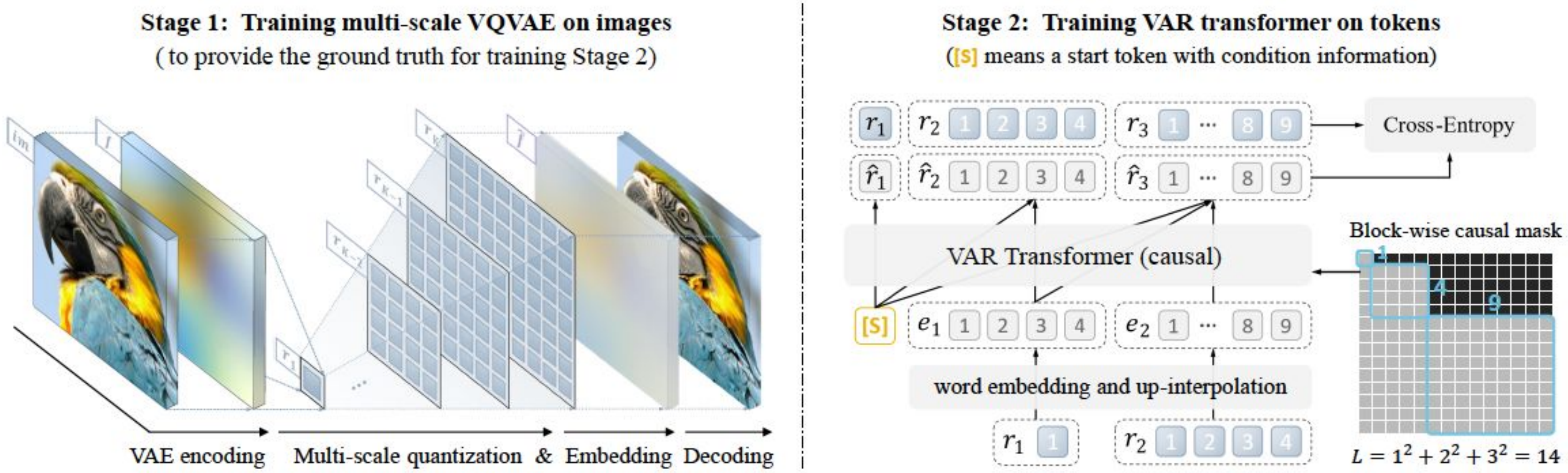
Autoregression assumes unidirectional token dependency:

$$p(x_1, x_2, \ldots, x_T) = \prod_{t=1}^{T} p(x_t \mid x_1, x_2, \ldots, x_{t-1})$$

But:

1. Feature map $f(i,j)$ has bi-directional correlations.
2. Neighboring tokens $q(i\pm1,j)$, $q(i,j\pm1)$ correlated with $q(i,j)$. Flattening disrupts spatial locality.
3. Patch-by-patch autoregressive steps are of $O(n^2)$.

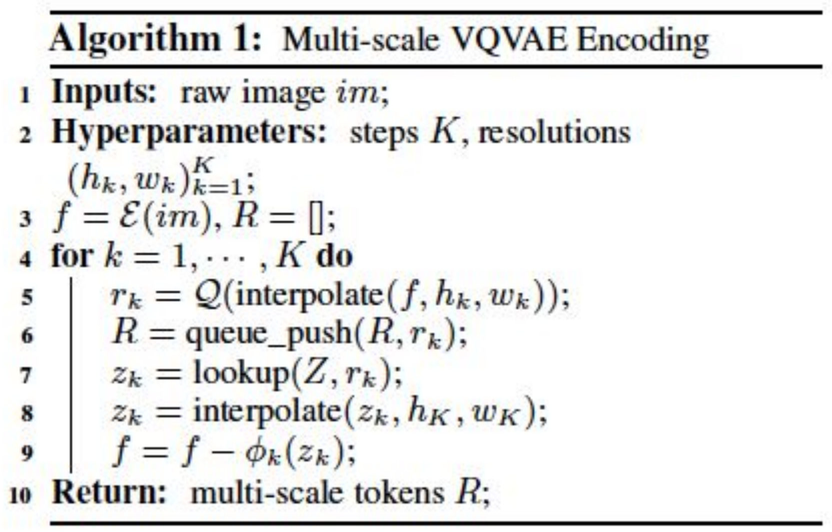
Visual autoregressive modeling - next scale prediction



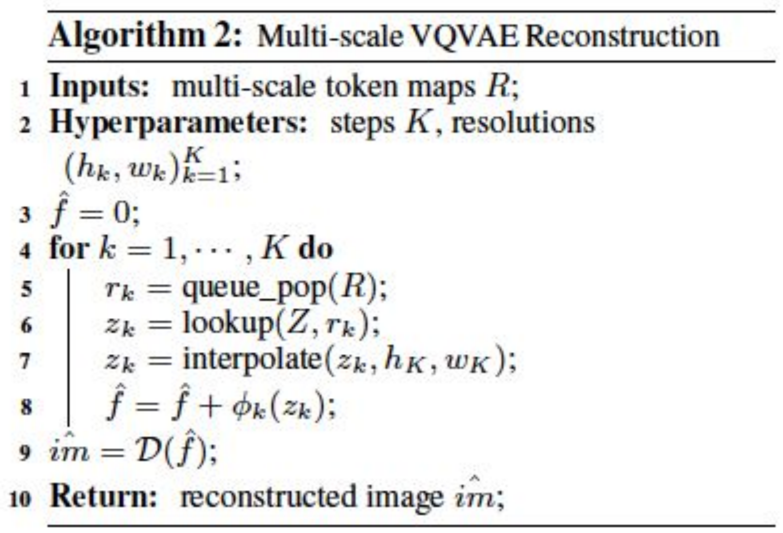
Stage 1: Training multi-scale VQVAE on images
(to provide the ground truth for training Stage 2)
im
f
r_K
r_{K-1}
r_{K-2}
r_1
$\hat{f}$
VAE encoding
Multi-scale quantization
&
Embedding
Decoding
Stage 2: Training VAR transformer on tokens
([S] means a start token with condition information)
Cross-Entropy
VAR Transformer (causal)
Block-wise causal mask
word embedding and up-interpolation
[S]
$L = 1^2 + 2^2 + 3^2 = 14$

Figure 4: **VAR involves two separated training stages. Stage 1:** a multi-scale VQ autoencoder encodes an image into K token maps $R = (r_1, r_2, \ldots, r_K)$ and is trained by a compound loss (5). For details on "Multi-scale quantization" and "Embedding", check Algorithm 1 and 2. **Stage 2:** a VAR transformer is trained via next-scale prediction (6): it takes $(\texttt{[s]}, r_1, r_2, \ldots, r_{K-1})$ as input to predict $(r_1, r_2, r_3, \ldots, r_K)$. The attention mask is used in training to ensure each r_k can only attend to $r_{\leq k}$. Standard cross-entropy loss is used.

Stage 1 - VQVAE training:



```
Algorithm 1: Multi-scale VQVAE Encoding
Inputs: raw image $im$;
Hyperparameters: steps $K$, resolutions $(h_k, w_k)_{k=1}^{K}$;
$f = \mathcal{E}(im)$, $R = []$;
for $k = 1, \cdots, K$ do
    $r_k = \mathcal{Q}(\text{interpolate}(f, h_k, w_k))$;
    $R = \text{queue\_push}(R, r_k)$;
    $z_k = \text{lookup}(Z, r_k)$;
    $z_k = \text{interpolate}(z_k, h_K, w_K)$;
    $f = f - \phi_k(z_k)$;
Return: multi-scale tokens $R$;
```

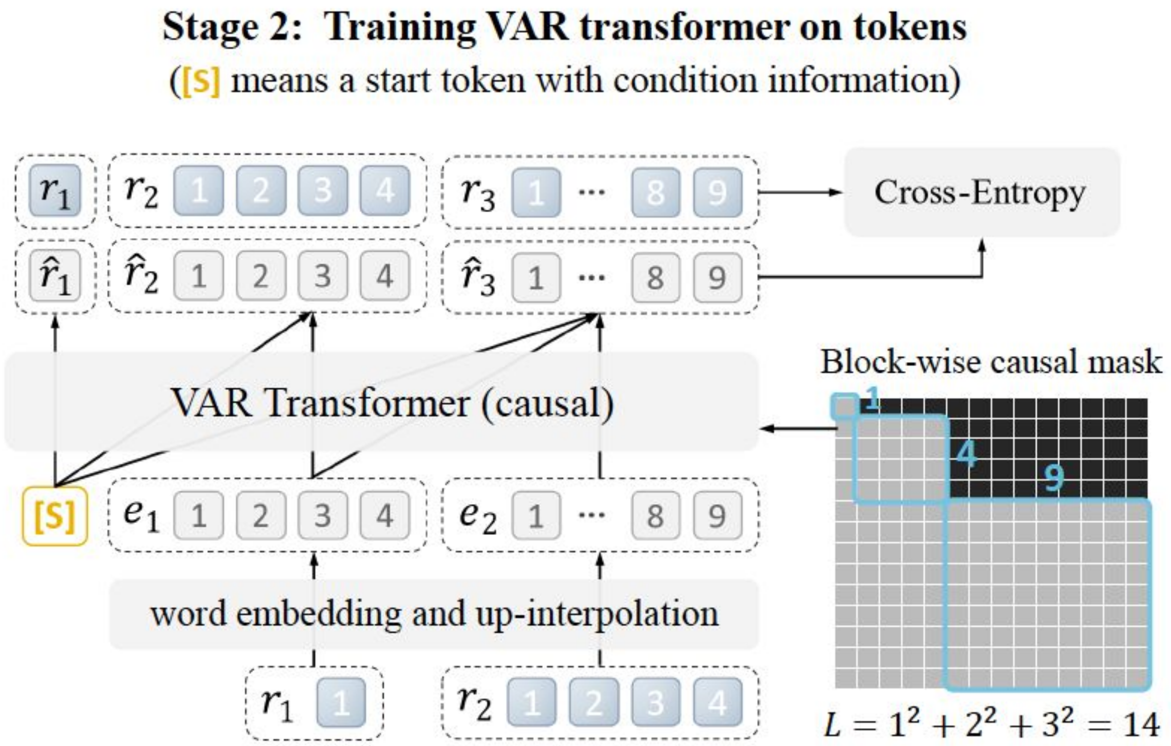


```
Algorithm 2: Multi-scale VQVAE Reconstruction
Inputs: multi-scale token maps $R$;
Hyperparameters: steps $K$, resolutions $(h_k, w_k)_{k=1}^{K}$;
$\hat{f} = 0$;
for $k = 1, \cdots, K$ do
    $r_k = \text{queue\_pop}(R)$;
    $z_k = \text{lookup}(Z, r_k)$;
    $z_k = \text{interpolate}(z_k, h_K, w_K)$;
    $\hat{f} = \hat{f} + \phi_k(z_k)$;
$\hat{im} = \mathcal{D}(\hat{f})$;
Return: reconstructed image $\hat{im}$;
```

Stage 2 - VAR transformer training for next-scale prediction conditioned on lower scale:

$$p(r_1, r_2, \ldots, r_K) = \prod_{k=1}^{K} p(r_k \mid r_1, r_2, \ldots, r_{k-1})$$

In each r_k, tokens are generated in parallel, reducing the $O(n^2)$ autogressive steps in next-token prediction approach.



Metric comparison

FID: compares distributions of generated and ground truth image sets

IS: quality of generated images (being distinct and diverse)

Precision: proportion of real-looking generated images

Recall: proportion of real images covered by generated images

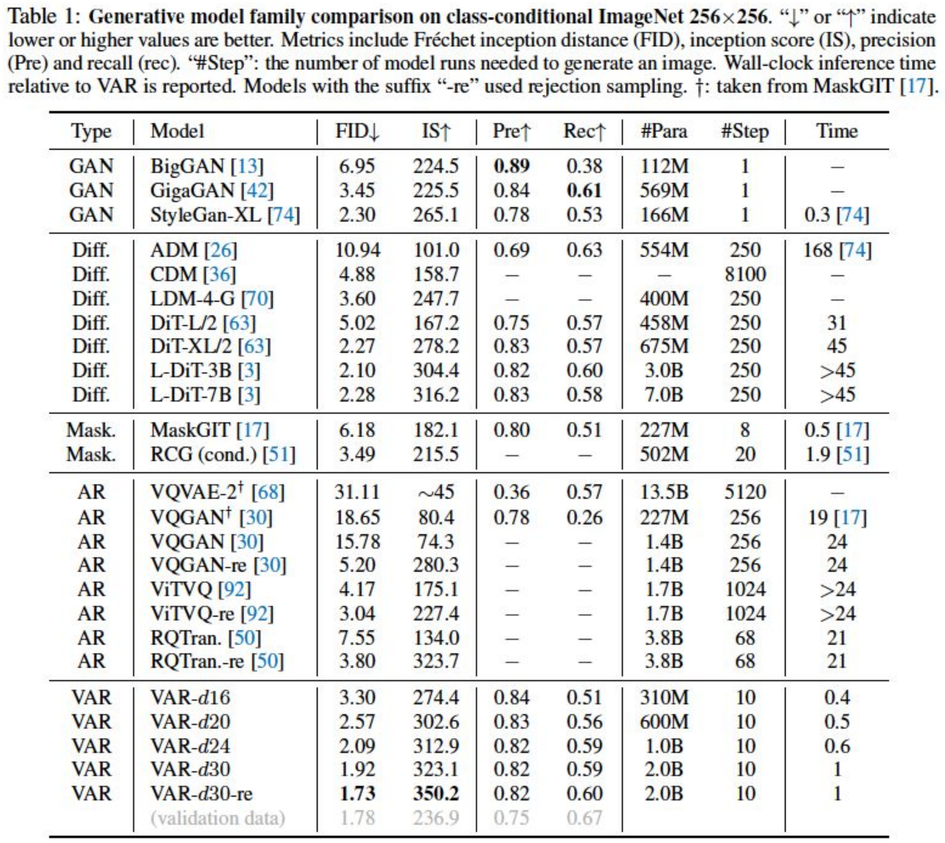


Table 1: **Generative model family comparison on class-conditional ImageNet 256×256.** "↓" or "↑" indicate lower or higher values are better. Metrics include Fréchet inception distance (FID), inception score (IS), precision (Pre) and recall (rec). "#Step": the number of model runs needed to generate an image. Wall-clock inference time relative to VAR is reported. Models with the suffix "-re" used rejection sampling. †: taken from MaskGIT [17].

Type	Model	FID↓	IS↑	Pre↑	Rec↑	#Para	#Step	Time
GAN	BigGAN [13]	6.95	224.5	**0.89**	0.38	112M	1	–
GAN	GigaGAN [42]	3.45	225.5	0.84	**0.61**	569M	1	–
GAN	StyleGan-XL [74]	2.30	265.1	0.78	0.53	166M	1	0.3 [74]
Diff.	ADM [26]	10.94	101.0	0.69	0.63	554M	250	168 [74]
Diff.	CDM [36]	4.88	158.7	–	–	–	8100	–
Diff.	LDM-4-G [70]	3.60	247.7	–	–	400M	250	–
Diff.	DiT-L/2 [63]	5.02	167.2	0.75	0.57	458M	250	31
Diff.	DiT-XL/2 [63]	2.27	278.2	0.83	0.57	675M	250	45
Diff.	L-DiT-3B [3]	2.10	304.4	0.82	0.60	3.0B	250	>45
Diff.	L-DiT-7B [3]	2.28	316.2	0.83	0.58	7.0B	250	>45
Mask.	MaskGIT [17]	6.18	182.1	0.80	0.51	227M	8	0.5 [17]
Mask.	RCG (cond.) [51]	3.49	215.5	–	–	502M	20	1.9 [51]
AR	VQVAE-2[†] [68]	31.11	~45	0.36	0.57	13.5B	5120	–
AR	VQGAN[†] [30]	18.65	80.4	0.78	0.26	227M	256	19 [17]
AR	VQGAN [30]	15.78	74.3	–	–	1.4B	256	24
AR	VQGAN-re [30]	5.20	280.3	–	–	1.4B	256	24
AR	ViTVQ [92]	4.17	175.1	–	–	1.7B	1024	>24
AR	ViTVQ-re [92]	3.04	227.4	–	–	1.7B	1024	>24
AR	RQTran. [50]	7.55	134.0	–	–	3.8B	68	21
AR	RQTran.-re [50]	3.80	323.7	–	–	3.8B	68	21
VAR	VAR-*d*16	3.30	274.4	0.84	0.51	310M	10	0.4
VAR	VAR-*d*20	2.57	302.6	0.83	0.56	600M	10	0.5
VAR	VAR-*d*24	2.09	312.9	0.82	0.59	1.0B	10	0.6
VAR	VAR-*d*30	1.92	323.1	0.82	0.59	2.0B	10	1
VAR	VAR-*d*30-re	**1.73**	**350.2**	0.82	0.60	2.0B	10	1
	(validation data)	1.78	236.9	0.75	0.67			

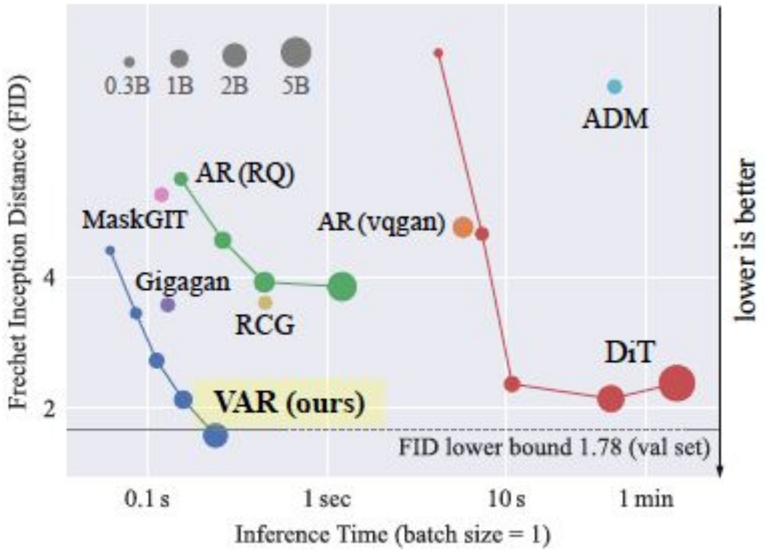


Figure 3: **Scaling behavior** of different model families on ImageNet 256×256 generation benchmark. The FID of the validation set serves as a reference lower bound (1.78). VAR with 2B parameters reaches an FID of 1.73, surpassing L-DiT with 3B or 7B parameters.

Scalability

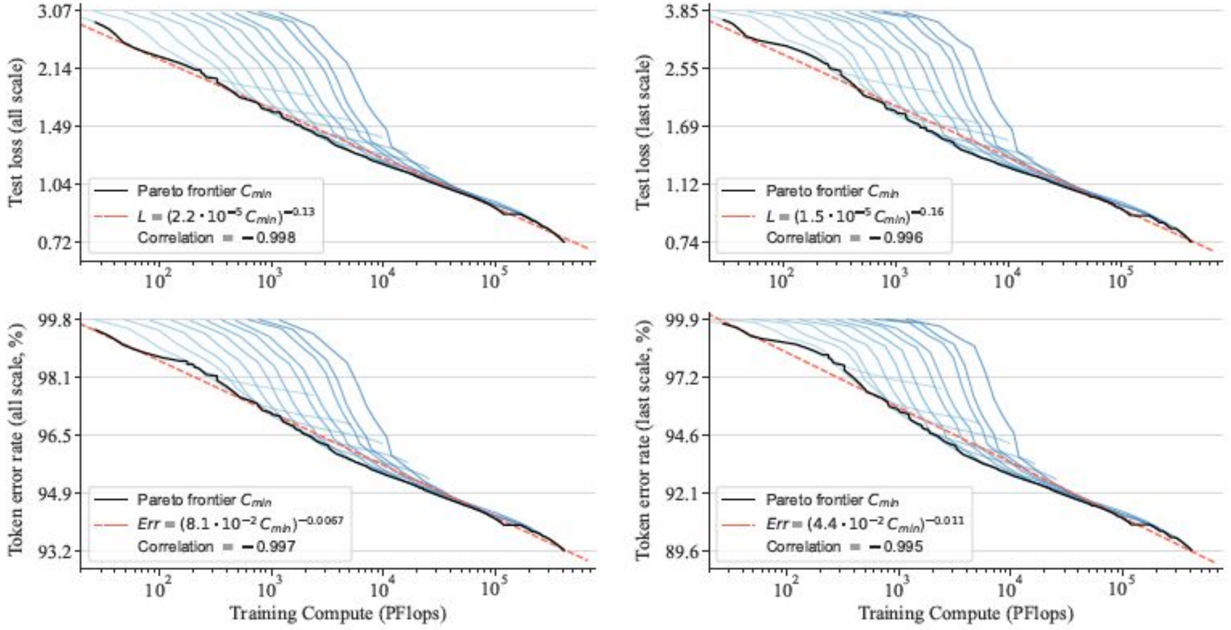
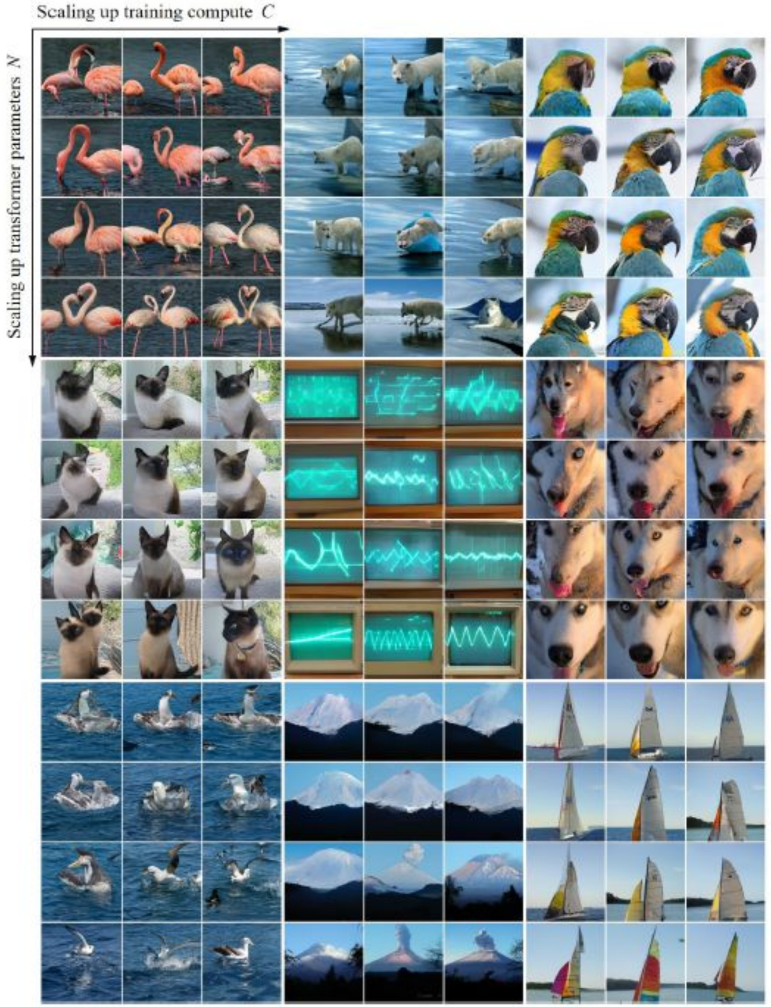
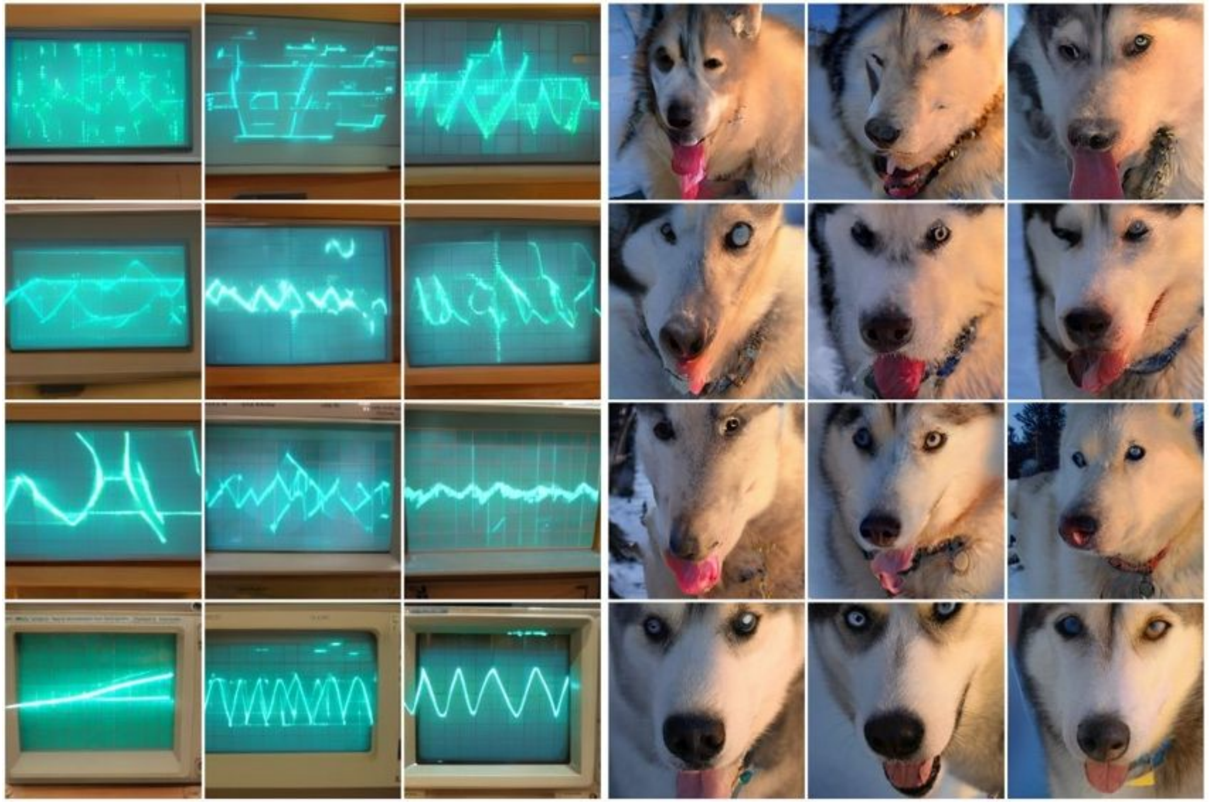


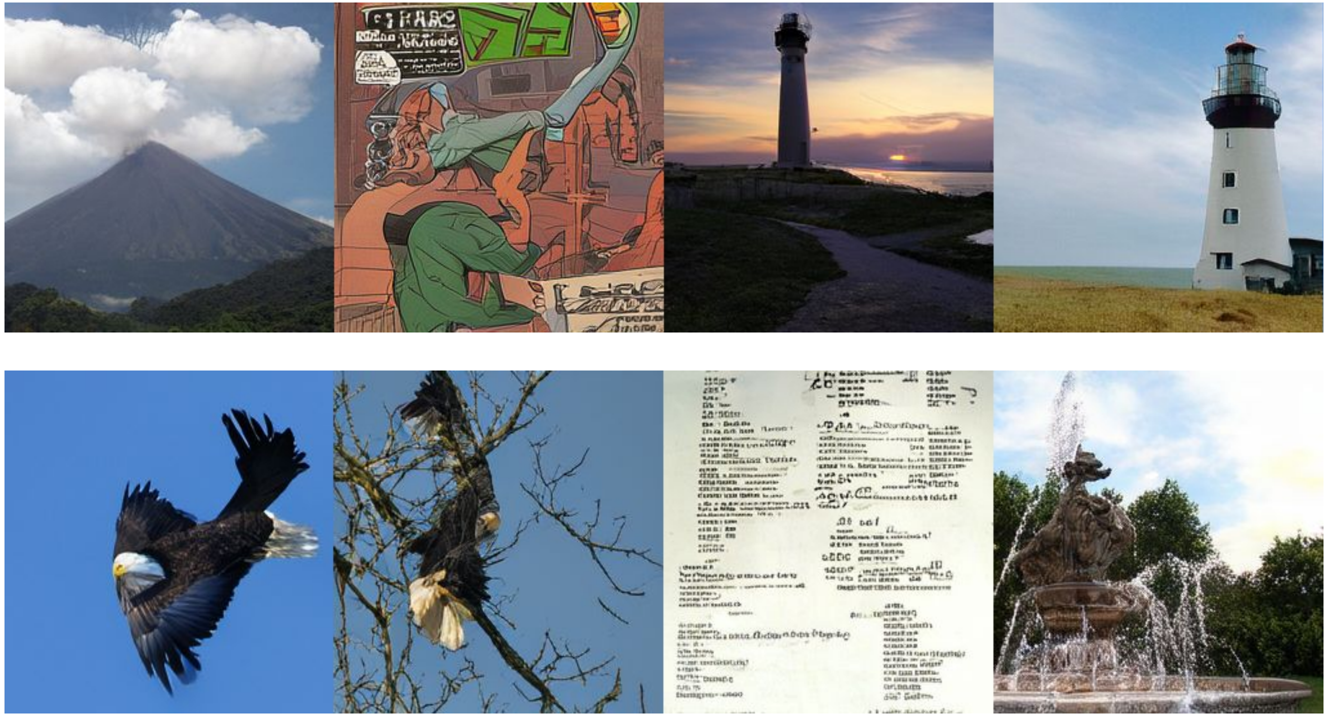
Figure 6: **Scaling laws with optimal training compute C_{min}.** Line color denotes different model sizes. Red dashed lines are power-law fits with equations in legend. Axes are on a logarithmic scale. Pearson coefficients near -0.99 indicate strong linear relationships between $\log(C_{\text{min}})$ *vs.* $\log(L)$ or $\log(C_{\text{min}})$ *vs.* $\log(Err)$.


Scaling up training compute C
Scaling up transformer parameters N

Let’s zoom in…



Trial example with Colab



Zero-shot tasks

In- and out-painting: Model only generates tokens within mask, without class label info in the model.

Class-conditional editing: Model generates tokens within bounding box, conditional on class labels.

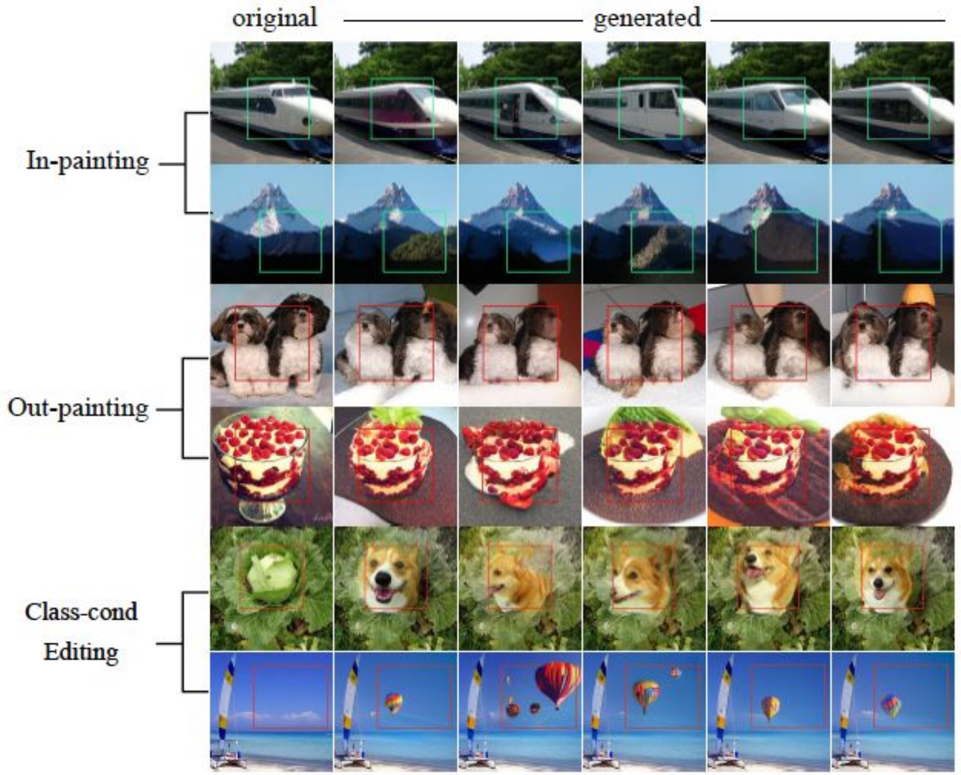


Figure 8: **Zero-shot evaluation in downstream tasks** containing in-painting, out-painting, and class-conditional editing. The results show that VAR can generalize to novel downstream tasks without special design and finetuning. Zoom in for a better view.

Trial example in Colab

Possible future extension

1. Text-prompt generation - integration with LLM
2. Video generation:
    - VAR can potentially handle longer temporal dependencies than diffusion model.
    - More efficient than traditional AR. More capable in generating high-resolution videos.