Logit Refiner: Improving Visual Autoregressive Models via Intra-Scale Dependency Modeling

Meimingwei Li[⋆,1], Stefan Andreas Baumann[⋆,1,2], Felix Krause[1,2], and Björn Ommer[1,2]

[1] CompVis @ LMU Munich, Germany
[2] Munich Center for Machine Learning (MCML)

Abstract. Visual Autoregressive Models (VAR) generate images through next-scale prediction, producing all tokens within each scale in parallel. We show that this parallel decoding constitutes a mean-field-style approximation that discards spatial dependencies among same-scale tokens, causing locally incoherent samples regardless of backbone capacity – a limitation of the *decoding rule*. Addressing this limitation, we introduce the *Logit Refiner*, a lightweight autoregressive module that restores intra-scale dependencies by sequentially sampling tokens conditioned on frozen backbone features. Adding only ∼10% parameters and less than 5% of the base model's training compute, it plugs into any pretrained VAR checkpoint without retraining. Controlled ablations isolate joint intra-scale sampling – rather than additional capacity or training – as the critical ingredient. Across backbones from 310M to 2B parameters on class-conditional ImageNet 256×256, the refiner consistently improves generation quality, enabling a 1.1B-parameter model to surpass one twice its size. The approach further generalizes to text-to-image generation, confirming that the mean-field bottleneck persists across VAR variants and is effectively alleviated by our method.
Project page: https://compvis.github.io/logit-refiner/.

1 Introduction

Autoregressive (AR) modeling achieves remarkable success in language [1–3, 5, 7, 14, 28, 68, 71, 74] by generating tokens sequentially from a learned joint distribution. Extending this paradigm to images, however, is challenging: images lack a canonical ordering [46, 50, 62, 79], and fully sequential pixel generation [46, 50, 62, 79] is prohibitively slow, as sequence lengths quickly reach tens of thousands of tokens.

Visual Autoregressive Modeling (VAR) [73] resolves this by leveraging *next-scale prediction*, generating images in a coarse-to-fine manner, while predicting all tokens within each scale in parallel. This design yields strong [61] generation results, scaling behavior, and substantially improved efficiency compared to token-wise autoregressive models.

⋆ Equal contribution.

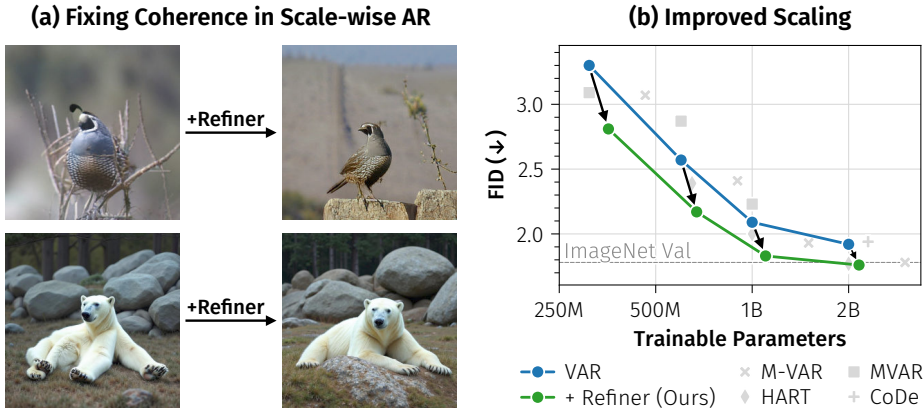


Fig. 1: **(a)** VAR [73] (top) and scaled-up variants like Infinity [24] (bottom) often generate spatially incoherent samples. Our Logit Refiner addresses this problem via a lightweight add-on to existing pretrained models. **(b)** The refiner consistently improves scaling behavior, shifting the VAR scaling curve downward, until saturating at the same FID that true unseen samples (validation set) achieve.

Despite accurate per-token predictions, VAR samples often exhibit *local spatial incoherence*: neighboring patches display mismatched textures, structural discontinuities, or implausible combinations – even when each individual prediction is plausible (Fig. 5). This raises a fundamental question:

If the per-token marginals are correct, why are the joint samples incoherent?

We identify the root cause as an implicit *mean-field-style assumption* in VAR's parallel within-scale decoding, which factorizes the conditional joint into independent per-token distributions, discarding spatial dependencies among tokens at the same scale. This approximation produces token combinations that are individually likely, yet jointly inconsistent. A minimal checkerboard example (Fig. 2) makes this concrete: correct per-pixel probabilities still yield invalid global patterns under independent sampling.

Crucially, this limitation lies in the *decoding rule* – even predicting the correct pointwise conditional distributions can not yield valid samples in practice, since tokens are decoded independently. The missing component in scale-wise autoregressive image generation is therefore *joint within-scale sampling* that accounts for the intra-scale token dependencies. We introduce the *Logit Refiner*, a lightweight *add-on* autoregressive module that restores intra-scale dependencies while leaving the VAR backbone untouched. It can directly be applied on top of an already pretrained VAR model, without any need for adaptation of the pretrained weights to obtain the quality benefits. Conditioned on the backbone's already-computed hidden states, the refiner samples tokens sequentially within each scale, requiring only a small causal model to capture the residual dependencies that independent decoding ignores. The module adds only ∼10% parameters, trains

in hours (<5% extra training compute) with the backbone frozen, and plugs into any pretrained VAR model with modest inference overhead.

Across backbone sizes from 310M to 2B parameters on class-conditional ImageNet, the Logit Refiner consistently improves generative performance by a wide margin, enabling VAR-d24 + Refiner (1.1B parameters) to surpass the twice as large VAR-d30 (2B). Controlled ablations confirm that these gains stem from *dependency modeling rather than additional capacity or training*: an architecture-matched refiner with bidirectional attention and independent sampling fails to match the autoregressive variant, isolating joint intra-scale sampling as the critical ingredient. The approach also generalizes to text-to-image generation, where the refiner yields consistent improvements on a 2B-parameter model.

Our work makes the following main contributions: starting by tracing the spatial incoherence observed in VAR samples to a specific cause – the mean-field-style approximation inherent in parallel within-scale decoding, which generates tokens independently regardless of the backbone's capacity – we show that autoregressive within-scale sampling is the minimal correction needed to remove this approximation error, reframing the problem from model capacity to the decoding rule. Then, we introduce a *lightweight* autoregressive refiner that implements this correction as a plug-in module over frozen backbone features, consistently improving generation quality across model scales from 310M to 2B parameters.

2 Related Work

Autoregressive Image Generation. Autoregressive models are a dominant paradigm for image generation, powering many frontier foundation models [21,22, 48,49,82]. Early approaches modeled images as pixel-level sequences [46,47,62,72], whereas modern methods [10, 32, 55, 65, 73, 84, 85] operate on discrete tokens from learned tokenizers [18,56,75,86] and explore alternatives to the standard "sweep" ordering [18,83,87], the grouping of multiple tokens [8, 58, 73, 80], or shared backbones with large language models [11,66,67,70,78,81]. Our work is orthogonal: rather than changing the token order or the backbone, we correct the independence assumption within parallel decoding groups.

Scale-wise Autoregressive Image Generation. VAR [73] introduced next-scale prediction, generating tokens at progressively finer resolutions while sampling all tokens within each scale in parallel. The paradigm has since been extended in many directions: Infinity and Switti scale it to text-to-image generation [24,41,76]; M-VAR [59] and MVAR [88] introduce more efficient backbones, and HMAR [31] combines next-scale prediction with masked autoregressive modeling [8]; FVAR [38], HART [69], and FlowAR [57] alter the prediction target or token representation; and a growing line of work reduces inference cost through token pruning [12,23,40], frequency- or entropy-guided skipping [9,34,39,89], and speculative decoding [13]. Beyond class-conditional generation, the paradigm has been applied to multimodal modeling [92,93], image editing [15,45], restora-

tion [54], super-resolution [53], segmentation [91], and video generation [29, 41]. Across these extensions, tokens within each scale remain at least partially decoded independently, suggesting that the mean-field-style approximation is a structural property of the scale-wise autoregressive paradigm rather than a task-specific limitation of VAR.

Refinement and Post-hoc Correction. Refining initial predictions appears in many forms, from draft-and-revise generation that iteratively improves masked tokens [33, 90] to image generators with explicit refinement stages that re-predict a full generated image [52, 77]. Other methods [cf. 4] propose adversarially optimizing guidance injection into the decoding rule to improve sample quality. Unlike these, we target a specific shortcoming of VAR-style models [73] – the mean-field-style approximation – and correct it with a lightweight add-on inside the generation loop, avoiding repeated full-model passes over the entire image.

3 Mean-Field-style Approximation in Scale Autoregression

Visual Autoregressive Modeling (VAR) [73] generates images through *next-scale prediction* over a hierarchy of discrete token maps. An image is encoded into K scales $\mathbf{r}_{1:K} = (\mathbf{r}_1, \ldots, \mathbf{r}_K)$, where each $\mathbf{r}_k \in [V]^{L_k}$ is a sequence of $L_k = h_k w_k$ tokens at spatial resolution $h_k \times w_k$. VAR models the joint distribution autoregressively *across scales*:

$$p(\mathbf{r}_{1:K}) = \prod_{k=1}^{K} p_\theta(\mathbf{r}_k \mid \mathbf{r}_{<k}). \tag{1}$$

At each scale, a transformer backbone f_θ produces hidden states $\mathbf{h}^{(k)} = f_\theta(\mathbf{r}_{<k})$ for all token positions in parallel that parametrize per-token categorical distributions. Tokens within each scale are then sampled *independently*:

$$p_\theta(\mathbf{r}_k \mid \mathbf{r}_{<k}) \approx \prod_{i=1}^{L_k} p_\theta\left(r_i^{(k)} \mid \mathbf{h}^{(k)}\right), \tag{2}$$

which constitutes a fully factorized (naive) *mean-field-style approximation* that ignores spatial dependencies across tokens in the same scale, akin to those typically used in variational inference in physics [30]. Natural images, however, exhibit strong spatial dependencies, implying that the true conditional joint $p_\theta(\mathbf{r}_k \mid \mathbf{r}_{<k})$ might *not* be modeled faithfully. This can lead to structurally incoherent samples that persist even at the largest model scales (Fig. 5).

Toy Example (Fig. 2). For a simple dataset containing only valid 2×2 checkerboards (two valid samples), a next-scale model predicts correct per-token marginals, yet independent sampling produces $2^4 = 16$ joint outcomes, most of which are invalid. This illustrates the central failure of Eq. (2): *correct marginals do not imply correct joint samples*, even when conditioning on previous scales.

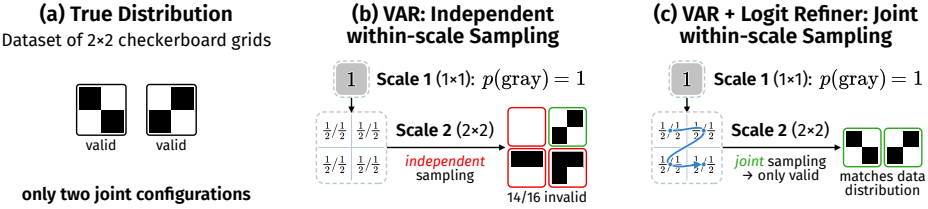


Fig. 2: Toy Example. (a) Consider a dataset of 2×2 checkerboards. **(b)** A VAR-style model can learn correct per-token marginals (gray at 1^2, $50/50$ at 2^2), yet independent sampling yields many invalid joint samples. **(c)** Our Logit Refiner models remaining dependencies autoregressively, restoring the joint and generating only valid samples.

This limitation arises from the *decoding rule*, motivating the restoration of the intra-scale joint distribution.

4 Logit Refiner

We introduce the *Logit Refiner*, a lightweight autoregressive module that restores intra-scale dependencies while leaving the pretrained VAR backbone unchanged. The backbone f_θ continues to compute per-token hidden states in parallel, whereas a separate refiner model q_ϕ models the joint distribution of tokens within each scale conditioned on these frozen features. Since the refiner operates purely at the decoding stage and is not constrained by VAR's mean-field-style factorization, it can implement flexible joint sampling with minimal additional parameters, computation, and training cost.

4.1 Restoring Joint Within-Scale Sampling

Since the backbone's bidirectional intra-scale attention already enables joint reasoning about all token positions per scale, the independence limitation resides in the *sampling rule*, not the learned representations (Sec. 3). We therefore replace the mean-field-style decoder in Eq. (2) with an autoregressive factorization over tokens within each scale:

$$q_\phi(\mathbf{r}_k \mid \mathbf{r}_{<k}) = \prod_{i=1}^{L_k} q_\phi\left(r_i^{(k)} \mid r_{<i}^{(k)}, \mathbf{h}_{\le i}^{(k)}, \mathbf{r}_{<k}\right), \tag{3}$$

where q_ϕ is a lightweight *refiner* model. Each token now conditions on both the frozen backbone features $\mathbf{h}_{\le i}^{(k)}$ and the previously sampled tokens $r_{<i}^{(k)}$, restoring the intra-scale joint that the parallel mean-field-style decoder discards. This formulation strictly generalizes the original decoder – setting the refiner in Eq. (3) to ignore the autoregressive context directly recovers Eq. (2) – making it a *minimal correction* that removes the conditional independence assumption without modifying the backbone.

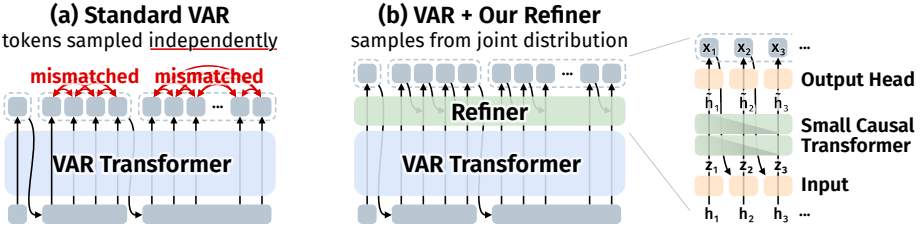


Fig. 3: Logit Refiner Overview. (a) The VAR backbone processes all previous scales and produces hidden states for the current scale in a single parallel forward pass, finally sampling from pointwise posteriors in parallel. As sampling is done independently within each scale, this can lead to mismatched tokens, affecting generation quality. **(b)** Our logit refiner takes these hidden states and samples tokens autoregressively within the scale, conditioning each prediction on previously sampled tokens. The refiner is a lightweight causal transformer, incurring only a small overhead during generation, while significantly improving sample quality.

Combining Eq. (3) with the scale-wise factorization Eq. (1) yields the full model:

$$p(\mathbf{r}_{1:K}) \approx \prod_{k=1}^{K} q_\phi(\mathbf{r}_k \mid \mathbf{r}_{<k}), \tag{4}$$

which preserves VAR's efficient across-scale generation while restoring within-scale dependencies. Unlike a fully autoregressive model that runs the entire backbone per token, only the lightweight refiner q_ϕ operates sequentially on a token level – the expensive backbone computation remains fully parallel. Because the refiner operates on the backbone's features rather than building context from scratch, it only needs to model *residual* dependencies, explaining why an extremely small model suffices (Sec. 4.2).

4.2 Architecture

The refiner is designed as a strict add-on: it never re-encodes the image and only consumes i) the already-computed backbone hidden states $\mathbf{h}^{(k)}$ for the current scale, and ii) the previously generated tokens *within* the current chunk $r^{(k)}_{<i}$. This separation ensures that the expensive backbone forward pass remains fully parallel; only the lightweight refiner runs sequentially within each scale. We describe each component below and illustrate the architecture in Fig. 3.

Inputs. For each position i in scale k, the refiner constructs an input vector from two streams of information: the frozen backbone hidden state $\mathbf{h}^{(k)}_i \in \mathbb{R}^w$, and an autoregressive context embedding $\mathbf{c}^{(k)}_i \in \mathbb{R}^w$ derived from the previously sampled token:

$$\mathbf{c}^{(k)}_i = \begin{cases} \mathbf{z}_{\text{sos}}, & i = 1, \\ \text{emb}_\phi(r^{(k)}_{i-1}), & i > 1, \end{cases} \qquad \mathbf{z}^{(k)}_i = \mathbf{W}_{\text{proj}}[\,\mathbf{h}^{(k)}_i \parallel \mathbf{c}^{(k)}_i\,], \tag{5}$$

with learned $\mathbf{z}_{\text{sos}}$, token embedding $\text{emb}_\phi : [V] \to \mathbb{R}^w$, and input projection $\mathbf{W}_{\text{proj}}$, and $[\,\cdot \,\|\, \cdot\,]$ denoting concatenation.

Refiner Blocks. A small stack of d_r transformer blocks processes the fused representations $\mathbf{z}^{(k)}_{1:L_k}$ with a causal mask, producing refined hidden states

$$\tilde{\mathbf{h}}^{(k)}_i = \text{TransformerBlocks}_\phi\big(\mathbf{z}^{(k)}_{1:i}\big). \tag{6}$$

Each block follows standard transformer design, matching the backbone's block architecture. Crucially, only $d_r \ll d$ blocks are needed (e.g., $d_r = 2$) – far fewer than the backbone depth ($d \in [16, 30]$), since the refiner already receives a rich, spatially-contextualized hidden state $\mathbf{h}^{(k)}_i$ from the backbone with bidirectional attention. The refiner only needs to model the *residual* dependencies not captured by the backbone – a much easier task than building spatial context from scratch.

The autoregressive factorization in Eq. (3) requires choosing a token ordering within each scale; we use standard raster-scan order (left-to-right, top-to-bottom), following conventions in patch-level autoregressive models [cf. 46, 47].

Output. An output head predicts logits from the refined hidden state, from which a token is sampled:

$$\tilde{\ell}^{(k)}_i = \text{head}_\phi\big(\tilde{\mathbf{h}}^{(k)}_i\big), \qquad r^{(k)}_i \sim \text{Cat}\big(\text{softmax}\big(\tilde{\ell}^{(k)}_i\big)\big). \tag{7}$$

The sampled token $r^{(k)}_i$ is then fed back as the autoregressive context for the next position via emb_ϕ in Eq. (5), and this process repeats sequentially for all L_k positions in the scale.

4.3 Training and Inference

Starting from a conventionally pretrained VAR model, we train the refiner with teacher forcing: during training, the autoregressive context $r^{(k)}_{<i}$ in Eq. (5) is replaced with ground-truth tokens. Combined with the causal attention mask, this allows training to be fully parallelized across all positions and scales simultaneously. We minimize cross-entropy over all tokens:

$$\mathcal{L}(\phi) = -\sum_{k=1}^{K}\sum_{i=1}^{L_k} \log q_\phi\Big(r^{(k)}_i \mid r^{(k)}_{<i},\, \mathbf{h}^{(k)}_{\le i},\, \mathbf{r}_{<k}\Big)\,. \tag{8}$$

During refiner training, the VAR backbone f_θ remains *frozen*. We only optimize the refiner parameters ϕ, finding that this suffices for achieving significant performance gains while keeping training cost minimal.

Identity Initialization. We design an initialization scheme that makes the refiner reproduce the base model's predictions at the start of training, so that optimization focuses entirely on learning the residual corrections needed for joint

sampling. Specifically, we copy the pretrained output head and token embedding weights from the base model, providing the refiner with an already well-structured output space. The autoregressive context integration via $\mathbf{W}_{\text{proj}}$ (in Eq. (5)) is initially disabled by setting $\mathbf{W}_{\text{proj}} \leftarrow [\mathbf{I} \parallel \mathbf{0}]$; similarly, the output projections of each transformer block's self-attention and feedforward networks are zero-initialized to leave the initial hidden states unchanged. Under this scheme, the model at initialization is functionally equivalent to the original VAR, and the training signal drives the refiner to learn only the *difference* between independent and joint within-scale distributions. As shown in Fig. 4, identity initialization retains the base model's generation quality from the first iteration and converges significantly faster than standard random initialization.

Inference. At inference, VAR generates tokens scale-wise in a coarse-to-fine manner. For each scale, we compute $\mathbf{h}^{(k)}$ once in parallel using the expensive base model f_θ. Then, we sample tokens $r_1^{(k)}, \ldots, r_{L_k}^{(k)}$ sequentially using KV caching in the refiner. Since the refiner is much smaller than the base model ($d_r \ll d$), the additional cost is modest and significantly lower than traditional full-model tokenwise autoregressive sampling.

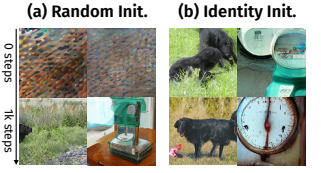


Fig. 4: Effect of Initialization Strategy.

5 Experiments

Experiment Settings. We conduct our experiments on class-conditional ImageNet [16] at a resolution of 256^2 unless noted otherwise, and primarily evaluate the Fréchet Inception Distance (FID) [25] on 50k generated samples. Our base model implementation and training setting directly follow VAR [73], except for a significantly reduced training duration of 30 epochs compared to multiple hundred for the base models, and a reduced base learning rate (1e-4 $\rightarrow$ 2.5e-5). Unless noted otherwise, we use $d_r = 2$ refiner layers whose width is matched to the base model's parameters. For sampling, we use classifier-free guidance (CFG) [27] and top-k sampling following the original VAR settings, with individually swept parameters. Full hyperparameters and details are provided in Supp. Sec. A.

5.1 Ablation Studies

We first validate the key design decisions of the Logit Refiner through controlled ablations, establishing that the gains originate from dependency modeling, before presenting final results.

Dependency Modeling, not Capacity, Drives Gains. Table 1 disentangles the contribution of joint intra-scale modeling from additional capacity and training on the same pretrained VAR-d16 backbone, exploring the following variations:

Table 1: What drives the gains? Neither additional training nor additional parallel layers match the improvement from autoregressive (joint) intra-scale refinement. All variants use VAR-d16 as backbone, the "parallel" and "AR" refiner use the same architecture ($d_r = 2$), with only a different attention mask and sampling. Only causal attention with autoregressive sampling – i.e., joint intra-scale modeling – yields substantial gains.

Model	Joint Modeling	Params	FID↓
VAR-d16 (baseline) [73]	✗	310M	3.30
+ Additional Training (30ep)	✗	310M	3.12
+ Parallel Refiner (bidirectional attention)	✗	356M	3.15
+ **AR Refiner (causal attention, ours)**	✓	356M	2.81

Table 2: Refiner Design Ablation. We start from VAR-d16 [73]. **(a)** Refiner depth: $d_r = 0$ (no transformer blocks) already improves FID significantly; $d_r = 2$ saturates quality. **(b)** Trainable components: the refiner alone captures most gains; jointly training the backbone adds little at much higher cost. **(c)** Per-scale importance: removing the refiner from any single scale degrades quality, most strongly at early scales; degradations diminish at high resolutions.

(a) Refiner Depth

Refiner Depth (d_r)	Params	FID↓
–	310M	3.30
0	+8M	3.02
1	+27M	2.85
2	+46M	2.81
4	+84M	2.82
8	+160M	2.81

(b) Trainable Components

Trained?			Trainable Params	FID↓
Output Head	Embedding	Base Backbone		
Baseline			310M	3.30
✓	✗	✗	46M	2.86
✗	✓	✗	40M	2.85
✓	✓	✗	46M	2.81
✓	✓	✓	356M	2.72

(c) Scales with Refiner

Scales	FID↓	Scales	FID↓
all	2.81	all \ $\{6^2\}$	2.85
all \ $\{2^2\}$	2.98	all \ $\{8^2\}$	2.81
all \ $\{3^2\}$	2.90	all \ $\{10^2\}$	2.85
all \ $\{4^2\}$	2.86	all \ $\{13^2\}$	2.86
all \ $\{5^2\}$	2.86	all \ $\{16^2\}$	2.83
[ctd. →]		–	3.30

1. **Additional Training:** continue training the base model for 30 more epochs without any architectural changes.
2. **Parallel Refiner:** add refiner-sized transformer blocks with bidirectional attention to the frozen backbone, matching our method's architecture and parameter count but sampling all tokens independently per scale, isolating capacity from joint modeling.
3. **AR Refiner (Ours):** identical architecture with causal attention and autoregressive sampling, restoring intra-scale dependencies.

All three variants are trained for the same number of epochs. Only the full AR refiner yields substantial gains, confirming that *dependency modeling* – not capacity or extra training – is the critical ingredient.

Small Refiners Suffice. Table 2a varies the number of refiner transformer blocks (d_r). Even a depth-0 refiner (no additional transformer blocks, just the autoregressive input projection and finetuned head with AR sampling) provides a meaningful improvement. This demonstrates that the autoregressive factorization *itself* is valuable, even when combined with just a causal context of one token

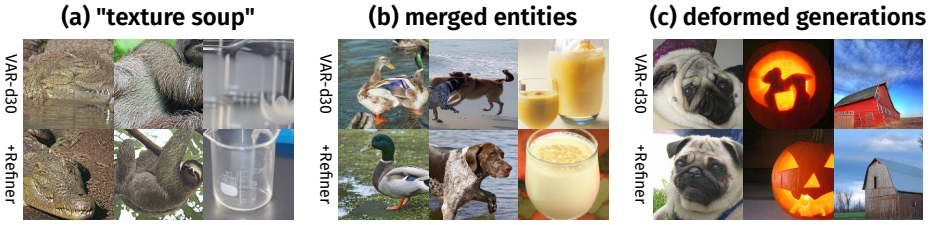


Fig. 5: VAR Failure Cases vs. Refiner. Our Logit Refiner can address a range of typical failures of VAR. Each column shows a paired sample (same class, same seed). Top: samples covering various failure modes from VAR-d30 [73]. Bottom: with refiner.

and a single additional linear layer to incorporate extra information. Performance saturates quickly, with $d_r = 2$ providing a favorable tradeoff.

Trainable Components. Table 2b ablates which components need to be trained. Reusing the frozen input embedding or output head from the base VAR model provides a small performance regression compared to training the whole refiner jointly, indicating that the backbone's learned representations already carry the relevant information – the refiner merely needs to model the residual dependencies that independent sampling discards. Jointly training the VAR backbone yields only minor further gains (FID 2.72) at greatly increased training cost, so we keep the backbone frozen throughout.

Which Scales Benefit Most? Table 2c measures each scale's contribution by applying the refiner at all but one scale during sampling. The FID degradation relative to full-refiner sampling quantifies how much joint modeling at that scale matters for generation quality. The largest drops occur at the earliest scales. This is notable, as the refiner's computational cost is also lowest at these scales, suggesting that selectively applying the refiner only at early scales could reduce inference overhead with minimal quality loss (Sec. 5.2).

Collectively, these ablations confirm that the Logit Refiner's gains stem from restoring intra-scale dependencies, not from additional capacity or training, and that the module is robust across architectural choices, with efficient add-on training on a *frozen pretrained* backbone sufficient to capture significant gains.

5.2 Main Results

ImageNet Generation. We identify three recurring failure modes of VAR that persist even at multi-billion parameter scales (Fig. 5-top): samples that consist of class-relevant textures with little obvious structure ("texture soup"), images with multiple inconsistent, often merged instances of the target class, and samples with locally inconsistent structure. All three stem from the lack of spatial coordination inherent in independent within-scale sampling (Sec. 3).

Table 3: System-level Comparison on class-conditional ImageNet-256^2 across discrete-token scale-wise autoregressive models. Within each backbone scale, VAR + Refiner achieves the best FID, improving by 0.16 to 0.49 over VAR with only ∼10% additional parameters. See Supp. Sec. B for additional models & evals w/o CFG.

Method	Params	FID↓	IS↑	Prec↑	Rec↑
MVAR-d16 [88]	310M	3.09	285.5	0.85	0.51
M-VAR-d16 [59]	464M	3.07	294.6	0.84	0.53
HMAR-d16 [31]	465M	3.01	288.6	0.84	0.55
VAR-d16 [73]	310M	3.30	274.4	0.84	0.51
+ Refiner (Ours)	356M	**2.81**$_{\blacktriangledown 0.49}$	267.2	0.81	0.56
HART-d20 [69]	649M	2.39	316.4	–	–
MVAR-d20 [88]	600M	2.87	295.3	0.86	0.52
M-VAR-d20 [59]	900M	2.41	308.4	0.85	0.58
HMAR-d20 [31]	840M	2.50	319.0	0.85	0.57
VAR-d20 [73]	600M	2.57	302.6	0.83	0.56
+ Refiner (Ours)	671M	**2.17**$_{\blacktriangledown 0.40}$	274.7	0.80	0.60
ImageNet Validation	–	1.78	–	–	–

Method	Params	FID↓	IS↑	Prec↑	Rec↑
HART-d24 [69]	1.0B	2.00	331.5	–	–
FastVAR-d24 [23]	1.0B	2.64	287.4	0.80	0.58
MVAR-d24 [88]	1.0B	2.23	300.1	0.86	0.52
M-VAR-d24 [59]	1.5B	1.93	320.7	0.83	0.59
HMAR [31]	1.3B	2.10	319.0	0.83	0.60
VAR-d24 [73]	1.0B	2.09	312.9	0.83	0.57
+ Refiner (Ours)	1.1B	**1.83**$_{\blacktriangledown 0.26}$	288.2	0.79	0.63
HART-d30 [69]	2.0B	1.77	330.3	–	–
FastVAR-d30 [23]	2.0B	2.30	288.7	0.81	0.59
VAR-CoDe-d30 [13]	2.3B	1.94	296	0.81	0.60
HMAR [31]	2.4B	1.95	334.5	0.82	0.62
VAR-d30 [73]	2.0B	1.92	323.1	0.82	0.58
+ Refiner (Ours)	2.2B	**1.76**$_{\blacktriangledown 0.16}$	319.4	0.80	0.62
M-VAR-d32 [59]	3.0B	1.78	331.2	0.83	0.61

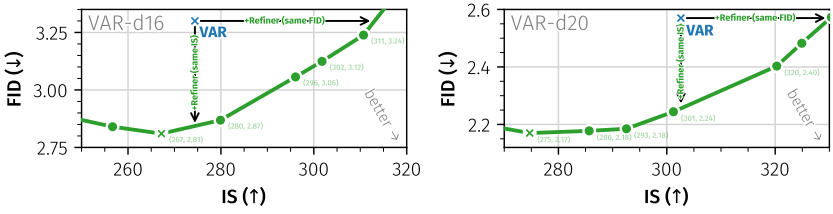


Fig. 6: FID-IS Improvement Tradeoff. By varying the classifier-free guidance scale, the logit refiner can achieve improvements in both dimensions.

Adding our refiner enables the model to sample from the joint intra-scale token distribution, directly addressing these failure modes (Fig. 5-bottom). We show additional qualitative samples in Supp. Sec. C.2.

These qualitative gains are reflected in quantitative metrics. Table 3 compares our Logit Refiner applied to VAR across scales and with a broad range of scale-wise autoregressive methods. Reference results for other generative model families are reported in the extended comparison (Supp. Tab. B.4). Across all model scales, the refiner consistently improves FID by a significant margin (0.16 to 0.49), while only adding ∼10% additional parameters. VAR-d24 + Refiner (1.1B params total) even exceeds VAR-d30 (2B params) by a significant margin, obtaining a stronger model at roughly half the size. Beyond FID, the refiner also consistently improves recall by 0.04 to 0.06 across all backbone scales, indicating that restoring intra-scale dependencies recovers sample diversity rather than trading it away. The small accompanying decrease in Inception Score follows from the lower CFG scales that are FID-optimal for the refiner, not from reduced sample quality – by varying the guidance scale, improvements in FID and/or IS over the baseline can be traded off (see Fig. 6). We also compare in a CFG-free

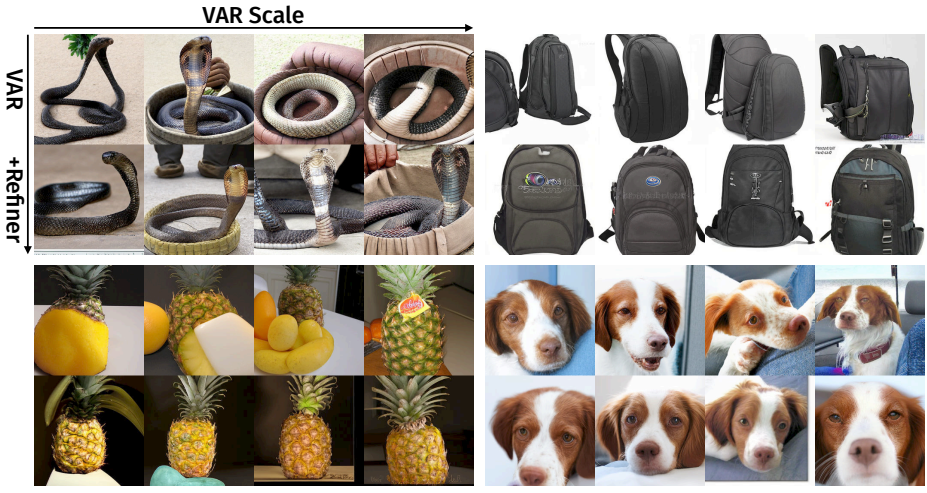
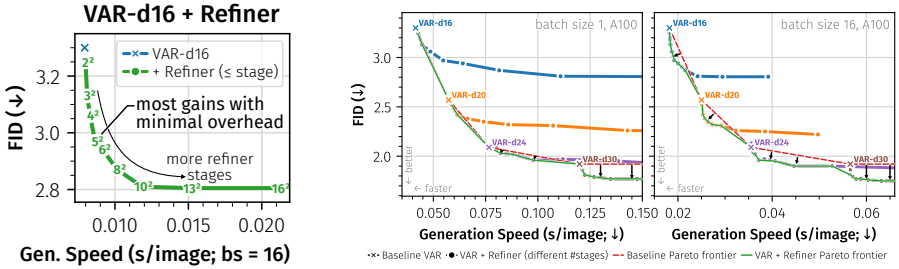


Fig. 7: Qualitative Scaling Behavior. Paired samples (same class, seed) across backbone scales (VAR-{16, 20, 24, 30}). *Top rows:* vanilla VAR improves visual fidelity and class consistency with scale, but spatial consistency problems persist at every size. *Bottom rows:* adding our refiner resolves these artifacts across all scales.

setting in Supp. Tab. B.5, where the logit refiner also consistently outperforms the baseline. Compared with other VAR variants that address orthogonal aspects, the basic VAR model with our refiner achieves the best generative performance within each model scale. Our approach does not utilize any of the improvements introduced by these methods, which may lead to further gains in combination. These directions are left for future work.

Scaling Behavior. Figure 7 investigates how the Logit Refiner interacts with backbone scale. Qualitatively, (Fig. 7), spatial consistency problems persist across all vanilla VAR scales, even as visual fidelity improves with model size. The refiner resolves these artifacts at every scale, confirming that the underlying issue is the mean-field-style sampling rule rather than insufficient model capacity. Quantitatively (Fig. 1b), the refiner shifts the FID scaling curve downward across all backbone sizes without altering the overall scaling trend, reflecting the qualitative scaling findings, and demonstrating that correcting the mean-field-style approximation can be more parameter-efficient than scaling the backbone.

Training and Inference Efficiency. The Logit Refiner corrects a sampling-time approximation, modeling *residual* dependencies between tokens within each scale. This suggests that a refiner trained on a frozen backbone should already capture most of the achievable gains, since the marginals are correct and only the dependencies are missing. Our results confirm this: adding a refiner to a frozen backbone improves the FID from 3.30 to 2.81 with only 66 H200-h of additional training compute. Training the full model with an integrated refiner from scratch



(a) VAR-d16 stage tradeoff **(b)** Across backbone scales and batch sizes

Fig. 8: Quality/Efficiency Tradeoff. (a) For VAR-d16, applying the refiner only at the first k scales (Tab. 2c) traverses the quality/efficiency tradeoff. **(b)** The same effect across backbone scales (d{16,20,24,30}) and batch-size regimes. *Left:* latency-optimized regime (batch size 1, the worst case for the refiner); *Right:* typical regime (batch size 16). At batch size 16, the refiner Pareto-dominates the baseline at every scale; at batch size 1, it adds intermediate operating points at low depths and dominates from d24/d30.

yields a stronger FID of 2.57, but requires 1,845 H200-h – 28× more compute for the remaining third of improvement. Jointly finetuning the backbone occupies a middle ground (FID 2.72, 127 H200-h). The dominant effect is thus the correction of the mean-field-style factorization itself, achievable as a lightweight post-hoc addition to any pretrained VAR model without retraining the backbone.

During inference, the refiner introduces sequential within-scale sampling, but the cost is modest. The expensive backbone forward pass remains fully parallel: for each scale k, the backbone computes all hidden states $\mathbf{h}^{(k)}$ in a single pass. Only the lightweight refiner blocks ($d_r = 2$ blocks vs. $d \in [16, 30]$ backbone blocks) run sequentially, and we employ KV caching to avoid redundant computations across tokens within a scale.

The relative overhead remains modest across backbone sizes and batch sizes (Fig. 8b), since the expensive backbone still runs once per scale in parallel and only the two-layer refiner is sequential – the refiner does not turn VAR into a fully token-wise autoregressive model. At a typical batch size, the refiner improves the quality/efficiency Pareto frontier over vanilla VAR at *every* backbone scale; in the latency-optimized single-sample regime, it adds finer-grained operating points and dominates the baseline from VAR-d24 onward.

Moreover, the scales ablation (Tab. 2c) shows that the refiner's quality gains concentrate at early scales, which contain the fewest tokens. For VAR-d16, this enables a practical trade-off (Fig. 8a): applying the refiner only at the first few scales can substantially reduce the sequential sampling cost while retaining most of the quality improvement. Concretely, applying the refiner on scales up to $8^2/10^2$ reduces the refiner overhead by 84%/71% while retaining 88%/99% of the full FID improvement, respectively.

Scaling to T2I. To validate that the Logit Refiner generalizes beyond class-conditional ImageNet, we apply it to Infinity [24], a scaled VAR variant for

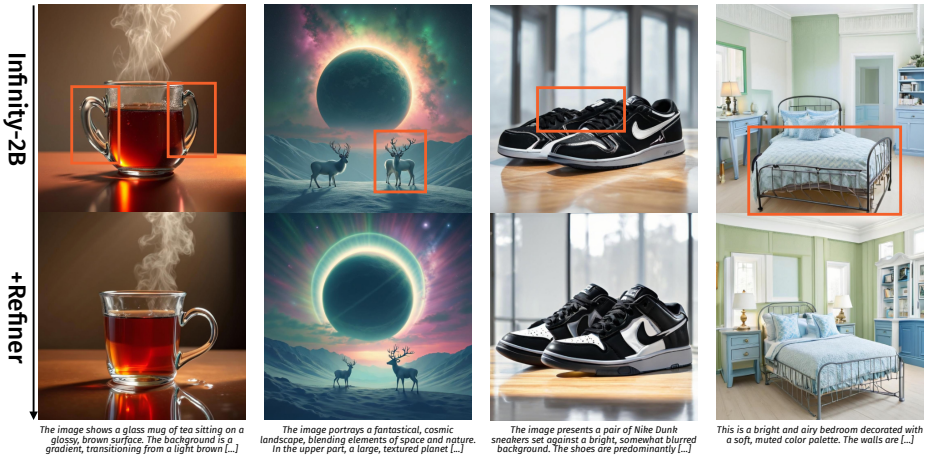


Fig. 9: Qualitative Text-to-Image Results. We show results from Infinity-2B [24] at 1024^2 resolution without (top) and with our refiner (bottom). As in the ImageNet class-conditional case, our Logit Refiner improves spatial coherence (inconsistencies in original images marked in **orange**) of the generated images.

Table 4: Quantitative T2I Results on HPSv3 [44]. We evaluate automated preference scores (higher is better) for samples generated with and without our Logit Refiner. It improves spatial consistency in generated images, reflected in better scores.

Model	Animals	Architecture	Arts	Characters	Design	Food	Natural Scenery	Others	Plants	Products	Science	Transportation	Average↑
Infinity-2B [24]	**9.57**	10.03	9.81	11.10	9.41	10.57	**9.08**	10.13	**10.14**	9.61	**8.57**	9.48	9.79
+ Refiner ($d_r = 2$)	9.53	**10.20**	**10.02**	**11.29**	**9.68**	**10.73**	9.04	**10.18**	**10.14**	**9.85**	8.50	**9.72**	**9.91**

text-to-image synthesis. We train the refiner on Infinity's backbone following a similar setup as for VAR (frozen backbone, $d_r = 2$, 100k steps at batch size 768; ∼640 H200-h train time – orders of magnitude less than the base model's pretraining time) on images and captions from FLUX-6M [19]. Following the findings from the previous paragraph, we apply the refiner selectively to the first several stages (up to resolution 6^2). Qualitatively (Fig. 9; see also Supp. Sec. C.1 for additional examples), the refiner yields the same types of improvements observed on ImageNet: improved structural coherence and reduced texture inconsistencies. These improvements are also reflected in quantitative evaluations: the refiner improves HPSv3 [44] scores from 9.79 to 9.91 (see Tab. 4), confirming that the benefits of restoring intra-scale dependencies transfer to open-vocabulary text-to-image generation.

6 Conclusion

Scale-wise visual autoregressive generation has a fundamental limitation: parallel within-scale decoding constitutes a mean-field-style approximation that discards spatial dependencies among tokens, causing locally incoherent samples regardless of backbone capacity or training duration. The Logit Refiner addresses this by restoring intra-scale dependencies through a lightweight autoregressive module that operates on parallel backbone features, adding only $\sim$10% parameters and requiring less than 5% of the base model's training compute. Across a large range of backbone sizes and both class- and text-conditional generation, the refiner consistently improves generation quality by a wide margin. Controlled ablations isolate joint intra-scale sampling rather than additional capacity or training as the critical ingredient.

Limitations and Future Work. Autoregressive within-scale sampling introduces sequential computation at each scale. While applying the refiner selectively at early scales retains the vast majority of quality gains at a fraction of the cost, the overhead is not fully eliminated. More broadly, our results suggest that when parallel decoding introduces mean-field-style assumptions, lightweight autoregressive correction can restore the discarded dependencies at minimal cost – a potentially general principle that shows promise for applications in parallel generative architectures [e.g. 8, 37]. Combining the refiner with orthogonal VAR improvements [e.g. 59, 69, 88] and exploring non-autoregressive approaches to within-scale sampling are further promising directions.

Acknowledgments

This project has been supported by the Horizon Europe project ELLIOT (GA No. 101214398), the project "GeniusRobot" (01IS24083) funded by the Federal Ministry of Research, Technology and Space (BMFTR), the BMWE ZIM-project (No. KK5785001LO4) "conIDitional LoRA", the German Federal Ministry for Economic Affairs and Energy within the project "NXT GEN AI METHODS - Generative Methoden für Perzeption, Prädiktion und Planung", and the bidt project KLIMA-MEMES. The authors gratefully acknowledge the Gauss Center for Supercomputing for providing compute through the NIC on JUWELS/JUPITER at JSC and the HPC resources supplied by the NHR@FAU Erlangen. We thank Ulrich Prestel, Jack Gallagher, Tommaso Martorella, Ming Gui, Nick Stracke, Kolja Bauer, and Vincent Tao Hu for feedback, proofreading, and helpful discussions, and Owen Vincent for technical support.

References

1. Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al.: Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023)

2. Anil, R., Dai, A.M., Firat, O., Johnson, M., Lepikhin, D., Passos, A., Shakeri, S., Taropa, E., Bailey, P., Chen, Z., et al.: Palm 2 technical report. arXiv preprint arXiv:2305.10403 (2023)
3. Bai, J., Bai, S., Chu, Y., Cui, Z., Dang, K., Deng, X., Fan, Y., Ge, W., Han, Y., Huang, F., et al.: Qwen technical report. arXiv preprint arXiv:2309.16609 (2023)
4. Bi, L., Huang, T., Guo, J., Xu, C.: Adversarial error correction for visual autoregressive generation (2026), https://arxiv.org/abs/2605.24843
5. Bi, X., Chen, D., Chen, G., Chen, S., Dai, D., Deng, C., Ding, H., Dong, K., Du, Q., Fu, Z., et al.: Deepseek llm: Scaling open-source language models with longtermism. arXiv preprint arXiv:2401.02954 (2024)
6. Brock, A., Donahue, J., Simonyan, K.: Large scale gan training for high fidelity natural image synthesis. arXiv preprint arXiv:1809.11096 (2018)
7. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. Advances in neural information processing systems **33**, 1877–1901 (2020)
8. Chang, H., Zhang, H., Jiang, L., Liu, C., Freeman, W.T.: Maskgit: Masked generative image transformer. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11315–11325 (2022)
9. Chen, J., Lin, R., Zheng, Z., Li, J., Li, M., Luo, G., et al.: Toprovar: Efficient visual autoregressive modeling via tri-dimensional entropy-aware semantic analysis and sparsity optimization. arXiv preprint arXiv:2602.22948 (2026)
10. Chen, M., Radford, A., Child, R., Wu, J., Jun, H., Luan, D., Sutskever, I.: Generative pretraining from pixels. In: III, H.D., Singh, A. (eds.) Proceedings of the 37th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 119, pp. 1691–1703. PMLR (13–18 Jul 2020)
11. Chen, X., Wu, Z., Liu, X., Pan, Z., Liu, W., Xie, Z., Yu, X., Ruan, C.: Janus-pro: Unified multimodal understanding and generation with data and model scaling. arXiv preprint arXiv:2501.17811 (2025)
12. Chen, Z., Fan, J., Yu, Z., Zhuang, B., Tan, M.: Frequency-aware autoregressive modeling for efficient high-resolution image synthesis. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 17140–17149 (2025)
13. Chen, Z., Ma, X., Fang, G., Wang, X.: Collaborative decoding makes visual autoregressive modeling efficient. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 23334–23344 (2025)
14. Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H.W., Sutton, C., Gehrmann, S., et al.: Palm: Scaling language modeling with pathways. Journal of machine learning research **24**(240), 1–113 (2023)
15. Dao, Q., He, X., Han, L., Nguyen, N.H., Nobar, A.H., Ahmed, F., Zhang, H., Nguyen, V.A., Metaxas, D.: Discrete noise inversion for next-scale autoregressive text-based image editing. arXiv preprint arXiv:2509.01984 (2025)
16. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 248–255 (2009). https://doi.org/10.1109/CVPR.2009.5206848
17. Dhariwal, P., Nichol, A.: Diffusion models beat gans on image synthesis. Advances in neural information processing systems **34**, 8780–8794 (2021)
18. Esser, P., Rombach, R., Ommer, B.: Taming transformers for high-resolution image synthesis. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 12873–12883 (2021)

19. Fang, R., Yu, A., Duan, C., Huang, L., Bai, S., Cai, Y., Wang, K., Liu, S., Liu, X., Li, H.: Flux-reason-6m & prism-bench: A million-scale text-to-image reasoning dataset and comprehensive benchmark. arXiv preprint arXiv:2509.09680 (2025)
20. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. Advances in neural information processing systems **27** (2014)
21. Google DeepMind, .: Gemini 2.5 Flash Image (Nano Banana). `https://deepmind.google/models/gemini-image/flash/` (2025)
22. Google DeepMind, .: Gemini 3 Pro Image (Nano Banana Pro). `https://deepmind.google/models/gemini-image/pro/` (2025)
23. Guo, H., Li, Y., Zhang, T., Wang, J., Dai, T., Xia, S.T., Benini, L.: Fastvar: Linear visual autoregressive modeling via cached token pruning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 19011–19021 (2025)
24. Han, J., Liu, J., Jiang, Y., Yan, B., Zhang, Y., Yuan, Z., Peng, B., Liu, X.: Infinity: Scaling bitwise autoregressive modeling for high-resolution image synthesis. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 15733–15744 (2025)
25. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems **30** (2017)
26. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Advances in neural information processing systems **33**, 6840–6851 (2020)
27. Ho, J., Salimans, T.: Classifier-free diffusion guidance. arXiv preprint arXiv:2207.12598 (2022)
28. Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Casas, D.d.L., Hendricks, L.A., Welbl, J., Clark, A., et al.: Training compute-optimal large language models. arXiv preprint arXiv:2203.15556 (2022)
29. Ji, L., Liu, X., Shang, J., Wang, S., Sun, Y., Wu, H., Wang, H.: Videoar: Autoregressive video generation via next-frame & scale prediction. arXiv preprint arXiv:2601.05966 (2026)
30. Jordan, M.I., Ghahramani, Z., Jaakkola, T.S., Saul, L.K.: An introduction to variational methods for graphical models. Machine learning **37**(2), 183–233 (1999)
31. Kumbong, H., Liu, X., Lin, T.Y., Liu, M.Y., Liu, X., Liu, Z., Fu, D.Y., Re, C., Romero, D.W.: Hmar: Efficient hierarchical masked auto-regressive image generation. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 2535–2544 (2025)
32. Lee, D., Kim, C., Kim, S., Cho, M., Han, W.S.: Autoregressive image generation using residual quantization. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11523–11532 (2022)
33. Lee, D., Kim, C., Kim, S., Cho, M., HAN, W.S.: Draft-and-revise: Effective image generation with contextual rq-transformer. Advances in Neural Information Processing Systems **35**, 30127–30138 (2022)
34. Li, J., Ma, Y., Zhang, X., Wei, Q., Liu, S., Zhang, L.: Skipvar: Accelerating visual autoregressive modeling via adaptive frequency-aware skipping. arXiv preprint arXiv:2506.08908 (2025)
35. Li, T., Chang, H., Mishra, S., Zhang, H., Katabi, D., Krishnan, D.: Mage: Masked generative encoder to unify representation learning and image synthesis. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2142–2152 (2023)
36. Li, T., He, K.: Back to basics: Let denoising generative models denoise. arXiv preprint arXiv:2511.13720 (2025)

37. Li, T., Tian, Y., Li, H., Deng, M., He, K.: Autoregressive image generation without vector quantization. Advances in Neural Information Processing Systems **37**, 56424–56445 (2024)
38. Li, X., Wu, C., Sun, Y., Zhou, J., Qu, D., Qu, Y., Bo, W., Yu, H., Liang, D.: Fvar: Visual autoregressive modeling via next focus prediction. arXiv preprint arXiv:2511.18838 (2025)
39. Li, Y., Wang, H., et al.: Freqexit: Enabling early-exit inference for visual autoregressive models via frequency-aware guidance. In: The Thirty-ninth Annual Conference on Neural Information Processing Systems (2025)
40. Li, Z., Wang, N., Bai, T., Mei, C., Wang, P., Qiu, S., Cheng, J.: Sparvar: Exploring sparsity in visual autoregressive modeling for training-free acceleration. arXiv preprint arXiv:2602.04361 (2026)
41. Liu, J., Han, J., Yan, B., Wuhui, Zhu, F., Wang, X., Jiang, Y., PENG, B., Yuan, Z.: Infinitystar: Unified spacetime autoregressive modeling for visual generation. In: The Thirty-ninth Annual Conference on Neural Information Processing Systems (2025)
42. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
43. Ma, N., Goldstein, M., Albergo, M.S., Boffi, N.M., Vanden-Eijnden, E., Xie, S.: Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In: European Conference on Computer Vision. pp. 23–40. Springer (2024)
44. Ma, Y., Wu, X., Sun, K., Li, H.: Hpsv3: Towards wide-spectrum human preference score. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 15086–15095 (2025)
45. Mao, Q., Cai, Q., Li, Y., Pan, Y., Cheng, M., Yao, T., Liu, Q., Mei, T.: Visual autoregressive modeling for instruction-guided image editing. arXiv preprint arXiv:2508.15772 (2025)
46. Van den Oord, A., Kalchbrenner, N., Espeholt, L., Vinyals, O., Graves, A., et al.: Conditional image generation with pixelcnn decoders. Advances in neural information processing systems **29** (2016)
47. Oord, A.v.d., Kalchbrenner, N., Kavukcuoglu, K.: Pixel recurrent neural networks. arXiv preprint arXiv:1601.06759 (2016)
48. OpenAI, .: Gpt image 1. `https://developers.openai.com/api/docs/models/gpt-image-1` (2025)
49. OpenAI, .: Gpt image 1.5. `https://developers.openai.com/api/docs/models/gpt-image-1.5` (2025)
50. Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, L., Shazeer, N., Ku, A., Tran, D.: Image transformer. In: International conference on machine learning. pp. 4055–4064. PMLR (2018)
51. Peebles, W., Xie, S.: Scalable diffusion models with transformers. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4195–4205 (2023)
52. Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., Rombach, R.: SDXL: Improving latent diffusion models for high-resolution image synthesis. In: The Twelfth International Conference on Learning Representations (2024)
53. Qu, Y., Yuan, K., Hao, J., Zhao, K., Xie, Q., Sun, M., Zhou, C.: Visual autoregressive modeling for image super-resolution. arXiv preprint arXiv:2501.18993 (2025)
54. Rajagopalan, S., Narayan, K., Patel, V.M.: Restorevar: Visual autoregressive generation for all-in-one image restoration. arXiv preprint arXiv:2505.18047 (2025)

55. Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., Sutskever, I.: Zero-shot text-to-image generation. In: International conference on machine learning. pp. 8821–8831. Pmlr (2021)
56. Razavi, A., Van den Oord, A., Vinyals, O.: Generating diverse high-fidelity images with vq-vae-2. Advances in neural information processing systems **32** (2019)
57. Ren, S., Yu, Q., He, J., Shen, X., Yuille, A., Chen, L.C.: Flowar: Scale-wise autoregressive image generation meets flow matching. arXiv preprint arXiv:2412.15205 (2024)
58. Ren, S., Yu, Q., He, J., Shen, X., Yuille, A., Chen, L.C.: Beyond next-token: Next-x prediction for autoregressive visual generation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 15781–15791 (2025)
59. Ren, S., Yu, Y., Ruiz, N., Wang, F., Yuille, A., Xie, C.: M-var: Decoupled scale-wise autoregressive modeling for high-quality image generation. arXiv preprint arXiv:2411.10433 (2024)
60. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10684–10695 (2022)
61. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International journal of computer vision **115**(3), 211–252 (2015)
62. Salimans, T., Karpathy, A., Chen, X., Kingma, D.P.: Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. arXiv preprint arXiv:1701.05517 (2017)
63. Sauer, A., Schwarz, K., Geiger, A.: Stylegan-xl: Scaling stylegan to large diverse datasets. In: ACM SIGGRAPH 2022 conference proceedings. pp. 1–10 (2022)
64. Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-based generative modeling through stochastic differential equations. arXiv preprint arXiv:2011.13456 (2020)
65. Sun, P., Jiang, Y., Chen, S., Zhang, S., Peng, B., Luo, P., Yuan, Z.: Autoregressive model beats diffusion: Llama for scalable image generation. arXiv preprint arXiv:2406.06525 (2024)
66. Sun, Q., Cui, Y., Zhang, X., Zhang, F., Yu, Q., Wang, Y., Rao, Y., Liu, J., Huang, T., Wang, X.: Generative multimodal models are in-context learners. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 14398–14409 (2024)
67. Sun, Q., Yu, Q., Cui, Y., Zhang, F., Zhang, X., Wang, Y., Gao, H., Liu, J., Huang, T., Wang, X.: Emu: Generative pretraining in multimodality. In: The Twelfth International Conference on Learning Representations (2024)
68. Sun, Y., Wang, S., Feng, S., Ding, S., Pang, C., Shang, J., Liu, J., Chen, X., Zhao, Y., Lu, Y., et al.: Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. arXiv preprint arXiv:2107.02137 (2021)
69. Tang, H., Wu, Y., Yang, S., Xie, E., Chen, J., Chen, J., Zhang, Z., Cai, H., Lu, Y., Han, S.: Hart: Efficient visual generation with hybrid autoregressive transformer. arXiv preprint arXiv:2410.10812 (2024)
70. Team, C.: Chameleon: Mixed-modal early-fusion foundation models. arXiv preprint arXiv:2405.09818 (2024)
71. Team, G., Anil, R., Borgeaud, S., Alayrac, J.B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A.M., Hauth, A., Millican, K., et al.: Gemini: a family of highly capable multimodal models. arXiv preprint arXiv:2312.11805 (2023)
72. Theis, L., Bethge, M.: Generative image modeling using spatial lstms. Advances in neural information processing systems **28** (2015)

73. Tian, K., Jiang, Y., Yuan, Z., Peng, B., Wang, L.: Visual autoregressive modeling: Scalable image generation via next-scale prediction. Advances in neural information processing systems **37**, 84839–84865 (2024)
74. Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al.: Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971 (2023)
75. Van Den Oord, A., Vinyals, O., et al.: Neural discrete representation learning. Advances in neural information processing systems **30** (2017)
76. Voronov, A., Kuznedelev, D., Khoroshikh, M., Khrulkov, V., Baranchuk, D.: Switti: Designing scale-wise transformers for text-to-image synthesis. arXiv preprint arXiv:2412.01819 (2024)
77. Wang, J., Zhou, Z., Mummadi, C.K., Dianat, S., Rabbani, M., Rao, R., Qiu, C., Tao, Z.: Visual self-refinement for autoregressive models. arXiv preprint arXiv:2510.00993 (2025)
78. Wang, X., Cui, Y., Wang, J., Zhang, F., Wang, Y., Zhang, X., Luo, Z., Sun, Q., Li, Z., Wang, Y., et al.: Multimodal learning with next-token prediction for large multimodal models. Nature pp. 1–7 (2026)
79. Wang, Y., Yang, S., Zhao, B., Zhang, L., Liu, Q., Zhou, Y., Xie, C.: Gpt-image-edit-1.5 m: A million-scale, gpt-generated image dataset. arXiv preprint arXiv:2507.21033 (2025)
80. Wang, Y., Ren, S., Lin, Z., Han, Y., Guo, H., Yang, Z., Zou, D., Feng, J., Liu, X.: Parallelized autoregressive visual generation. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 12955–12965 (2025)
81. Wu, C., Chen, X., Wu, Z., Ma, Y., Liu, X., Pan, Z., Liu, W., Xie, Z., Yu, X., Ruan, C., et al.: Janus: Decoupling visual encoding for unified multimodal understanding and generation. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 12966–12977 (2025)
82. xAI, .: Grok imagine. `https://docs.x.ai/developers/model-capabilities/images/generation` (2025)
83. Xu, Y., Ju, J., Luan, J., Cui, J.: Direction-aware diagonal autoregressive image generation. arXiv preprint arXiv:2503.11129 (2025)
84. Yu, J., Li, X., Koh, J.Y., Zhang, H., Pang, R., Qin, J., Ku, A., Xu, Y., Baldridge, J., Wu, Y.: Vector-quantized image modeling with improved vqgan. arXiv preprint arXiv:2110.04627 (2021)
85. Yu, J., Xu, Y., Koh, J.Y., Luong, T., Baid, G., Wang, Z., Vasudevan, V., Ku, A., Yang, Y., Ayan, B.K., et al.: Scaling autoregressive models for content-rich text-to-image generation. arXiv preprint arXiv:2206.10789 **2**(3), 5 (2022)
86. Yu, L., Lezama, J., Gundavarapu, N.B., Versari, L., Sohn, K., Minnen, D., Cheng, Y., Birodkar, V., Gupta, A., Gu, X., et al.: Language model beats diffusion–tokenizer is key to visual generation. arXiv preprint arXiv:2310.05737 (2023)
87. Yu, Q., He, J., Deng, X., Shen, X., Chen, L.C.: Randomized autoregressive visual generation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 18431–18441 (2025)
88. Zhang, J., Long, W., Han, M., You, W., Gu, S.: MVAR: Visual autoregressive modeling with scale and spatial markovian conditioning. In: The Fourteenth International Conference on Learning Representations (2026)
89. Zhang, Y., Liu, J., Liu, F., Miao, D., Zhang, Q., Fu, K., Wang, C., Cao, L.: Adaptive visual autoregressive acceleration via dual-linkage entropy analysis. arXiv preprint arXiv:2602.01345 (2026)
90. Zheng, H.K., Li, P.: Lsrs: Latent scale rejection sampling for visual autoregressive modeling. arXiv preprint arXiv:2512.03796 (2025)

91. Zheng, R., Qi, L., Chen, X., Wang, Y., Wang, K., Zhao, H.: Seg-var: Image segmentation with visual autoregressive modeling. arXiv preprint arXiv:2511.12594 (2025)
92. Zhuang, X., Xie, Y., Deng, Y., Liang, L., Ru, J., Yin, Y., Zou, Y.: Vargpt: Unified understanding and generation in a visual autoregressive multimodal large language model. arXiv preprint arXiv:2501.12327 (2025)
93. Zhuang, X., Xie, Y., Deng, Y., Yang, D., Liang, L., Ru, J., Yin, Y., Zou, Y.: Vargpt-v1. 1: Improve visual autoregressive large unified model via iterative instruction tuning and reinforcement learning. arXiv preprint arXiv:2504.02949 (2025)

Supplementary Material

A Implementation Details

Training Hyperparameters. We show relevant hyperparameters for all trained model variations in Supp. Table A.1. Optimizer settings have been directly taken from VAR, with the exception of a reduced learning rate (result of a sweep; we find that the learning rate does not meaningfully influence the FID once converged for our base configuration, with more than double and less than half the learning rate resulting in similar sample metrics) and a linear warmup + cosine decay schedule. For all VAR models, we train for 40 epochs. All variants are converged by that time. For Infinity [24], we train for 200k steps.

Table A.1: Hyperparameters.

Variant	Ablations	Main Results	Text-to-Image
Dataset	ImageNet-256^2	ImageNet-256^2	FLUX-6M [19]
Base Model	VAR-d16 [73]	VAR-d{16,20,24,30} [73]	Infinity-2B [24]
Base Model Depth d	16	{16,20,24,30}	32
Base Model Width	1024	{1024,1280,1536,1920}	2048
Refiner Depth d_r	{0, 1, 2, 4, 8}	2	2
Refiner Width	1024 (matching base)	matching base	2048 (matching base)
Trainable Parameters	{8M,27M,46M,84M,160M}	varying	193M
Batch Size	768	768 if $d < 30$, else 1024	768
Training Duration	30 epochs	30 epochs	200k steps
Precision	fp16 MP	fp16 MP	bf16 MP
Training Hardware	8 H200	8-32 H200	64 H200
Optimizer	AdamW [42]	AdamW [42]	AdamW [42]
Base LR ($LR = LR_{\text{base}} \cdot BS/256$)	$2.5 \cdot 10^{-5}$	$2.5 \cdot 10^{-5}$	$2.5 \cdot 10^{-5}$
Learning Rate Warmup	2% of training from 0.5% of peak	2% of training from 0.5% of peak	2% of training from 0.5% of peak
Learning Rate Schedule	linear	linear	linear
Betas (β_1, β_2)	$(0.9, 0.95)$	$(0.9, 0.95)$	$(0.9, 0.95)$
Weight Decay	0.05	0.05; d30: scheduled following VAR [73]	0.01

Inference Hyperparameters. We observe that, like with most other image generation models, final performance of VAR + Logit Refiner as measured by FID varies with inference hyperparameters, whose optimal values vary with base model size. We therefore sweep both top-k and CFG [27] scales individually w.r.t. FID. Over CFG, results are generally smooth (i.e., straightforward to sweep, typically close to convex). Supp. Figure A.1 shows the result of our hyperparameter search, where we sweep the CFG scale in increments of 0.1 and k in increments of 100 around the optimum. FID [25] is computed on 50k samples with class-balanced sampling following the baseline VAR [73]. The optimal params we found are listed in Supp. Tab. A.2.

Evaluation Details. For our VAR-d16+2 refiner, we try training with different random seeds to estimate confidence intervals for our results. Across four training runs, we obtain the following FIDs: 2.77, 2.80, 2.81, 2.83, resulting in a sample

Table A.2: Inference Hyperparameters. Used for quantitative evaluations. Parameters for Infinity-2B directly mirror those of the base model.

Base Model	VAR-d16	VAR-d20	VAR-d24	VAR-d30	Infinity-2B
CFG w	1.8	1.5	1.5	1.9	3.0
Top-k	1100	900	800	500	900
Top-p	–	–	–	–	0.97
τ	–	–	–	–	0.5

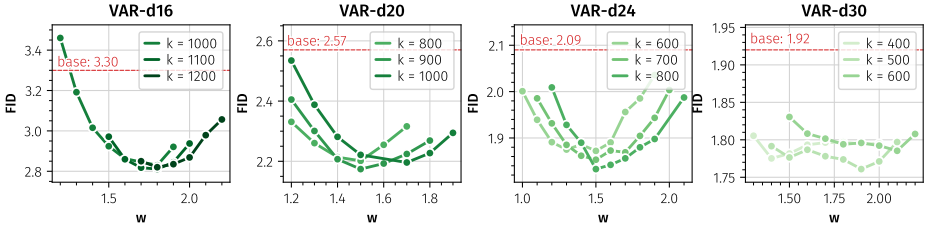


Fig. A.1: Inference Hyperparameter Exploration. The optimal classifier-free guidance scale w and top-k threshold for VAR with our refiner vary across scales.

standard deviation of 0.025. This puts our gains compared to all baselines in Tab. 3 to at least 8 standard deviations, indicating that the gains are statistically significant. Across all runs, we do not choose specific checkpoints, but use the same training setup, including seed. For HPSv3, we evaluate the first 50 prompts per subset (600 images total), since sampling all images would be prohibitively expensive.

Within-Scale Token Ordering. We generally use "sweep"/row-major token ordering for the refiner. We explore whether this choice is relevant for final performance by training refiners with five different intra-scale orderings in Supp. Tab. A.3. Inference hparams are reused from original "sweep" order; per-order tuning would likely close the small remaining differences. All tested orderings retain the gain, showing that the improvement is not an artifact of raster scan.

B Additional Quantitative Evaluation Details

We show an extended version of Tab. 3 in Supp. Tab. B.4. In addition to the major improvement in FID discussed in the main paper, we also observe a minor decrease in IS and precision, and an increase in recall. We attribute the reduction of IS values primarily to VAR with a refiner requiring lower CFG scales w for optimal FID (higher guidance scales are generally associated with higher IS values). Further, we show evaluations without classifier-free guidance in Supp. Tab. B.5: gains with the refiner vs. the baseline persist, and are more pronounced for large models than for small models.

Table A.3: Influence of Intra-Scale Token Ordering. All orderings yield near-identical gains.

Method	VAR-d16	+ Refiner (Ours)				
Sampling Order	parallel	sweep (paper)	col.-major	alternate	spiral in	spiral out
FID (↓)	3.30	2.81$_{\blacktriangledown 0.49}$	2.83$_{\blacktriangledown 0.47}$	2.82$_{\blacktriangledown 0.48}$	2.83$_{\blacktriangledown 0.47}$	2.86$_{\blacktriangledown 0.44}$

Table B.4: Extended System-level Comparison on class-conditional ImageNet-256^2 (extending Tab. 3).

Method	Params	FID↓	IS↑	Prec↑	Rec↑
Scale-wise Autoregression [73]					
MVAR-d16 [88]	310M	3.09	285.5	0.85	0.51
M-VAR-d16 [59]	464M	3.07	294.6	0.84	0.53
HMAR-d16 [31]	465M	3.01	288.6	0.84	0.55
VAR-d16 [73]	310M	3.30	274.4	0.84	0.51
+ Refiner (Ours)	356M	**2.81**$_{\blacktriangledown 0.49}$	267.2	0.81	0.56
HART-d20 [69]	649M	2.39	316.4	–	–
MVAR-d20 [88]	600M	2.87	295.3	0.86	0.52
M-VAR-d20 [59]	900M	2.41	308.4	0.85	0.58
HMAR-d20 [31]	840M	2.50	319.0	0.85	0.57
VAR-d20 [73]	600M	2.57	302.6	0.83	0.56
+ Refiner (Ours)	671M	**2.17**$_{\blacktriangledown 0.40}$	274.7	0.80	0.60
HART-d24 [69]	1.0B	2.00	331.5	–	–
FastVAR-d24 [23]	1.0B	2.64	287.4	0.80	0.58
MVAR-d24 [88]	1.0B	2.23	300.1	0.86	0.52
M-VAR-d24 [59]	1.5B	1.93	320.7	0.83	0.59
HMAR [31]	1.3B	2.10	319.0	0.83	0.60
VAR-d24 [73]	1.0B	2.09	312.9	0.83	0.57
+ Refiner (Ours)	1.1B	**1.83**$_{\blacktriangledown 0.26}$	288.2	0.79	0.63
HART-d30 [69]	2.0B	1.77	330.3	–	–
FastVAR-d30 [23]	2.0B	2.30	288.7	0.81	0.59
VAR-CoDe-d30 [13]	2.3B	1.94	296	0.81	0.60
HMAR [31]	2.4B	1.95	334.5	0.82	0.62
VAR-d30 [73]	2.0B	1.92	323.1	0.82	0.58
+ Refiner (Ours)	2.2B	**1.76**$_{\blacktriangledown 0.16}$	319.4	0.80	0.62
M-VAR-d32 [59]	3.0B	1.78	331.2	0.83	0.61
Generative Adversarial Nets [20]					
BigGAN-deep [6]	112M	6.95	202.6	0.87	0.28
StyleGAN-XL [63]	166M	2.30	265.1	0.78	0.53
***Diffusion* [26, 64]**					
ADM-G [17]	554M	4.59	186.7	0.82	0.52
LDM-4-G [60]	400M	3.60	247.6	–	–
DiT-XL/2 [51]	675M	2.27	278.2	0.83	0.57
SiT-XL/2 [43]	675M	2.06	252.2	–	–
JiT-G/16 [36]	2.0B	1.82	292.6	0.79	0.62
FlowAR [57]	1.9B	1.65	296.5	0.83	0.60
Raster Autoregression					
VQGAN [18]	1.4B	15.78	74.3	–	–
RQ-Transformer [33]	3.8B	7.55	134.0	0.73	0.58
LlamaGen-3B [65]	3.1B	2.18	263.3	0.81	0.58
RAR-XXL [87]	1.5B	1.48	326.0	0.80	0.63
Masked Autoregression					
MAGE [35]	439M	7.04	123.5	–	–
MaskGiT [8]	227M	6.18	182.1	0.80	0.51
MAR-H [37]	943M	1.55	303.7	0.81	0.62
ImageNet Validation	–	1.78	–	–	–

Table B.5: Extra Evaluations without CFG across Scales. Similar to the typical inference setting with CFG **(a)**, our refiner also enables significant gains in generative quality without it **(b)**.

FID (↓)	**(a)** CFG: ✓				**(b)** CFG: ✗			
	d16	d20	d24	d30	d16	d20	d24	d30
VAR	3.30	2.57	2.09	1.92	3.44	2.62	2.13	2.17
+ Refiner (Ours)	2.81$_{\blacktriangledown 0.49}$	2.17$_{\blacktriangledown 0.40}$	1.83$_{\blacktriangledown 0.26}$	1.76$_{\blacktriangledown 0.16}$	3.41$_{\blacktriangledown 0.03}$	2.40$_{\blacktriangledown 0.22}$	1.94$_{\blacktriangledown 0.19}$	1.88$_{\blacktriangledown 0.29}$

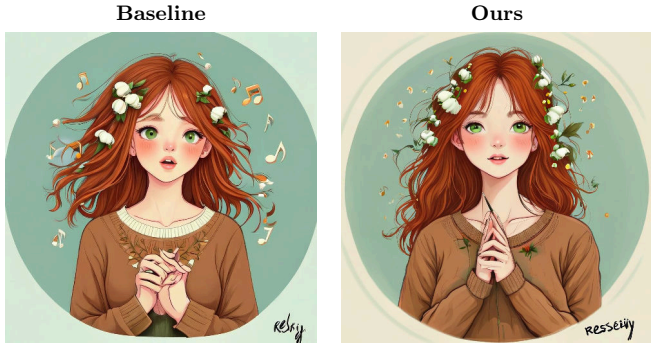
C Additional Qualitative Samples

C.1 Text-to-Image

We show additional comparisons between the base Infinity-2B [24] and our refiner version in Supp. Figs. C.2 and C.3.

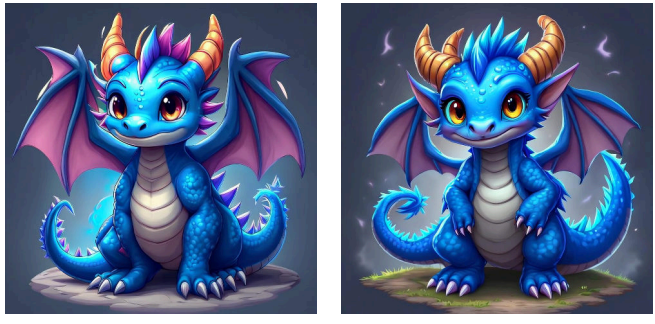
C.2 ImageNet

Supp. Figure C.4 shows additional results of VAR with and without our refiner across different base model scales. Supp. Figures C.5 to C.10 show uncurated ImageNet samples for various classes.



The image showcases a detailed and enchanting illustration of a young woman, possibly a sketch in progress, set against a light teal circular backdrop ...

The image features a young woman with a striking blend of modern and traditional Japanese-inspired aesthetics. She is captured from the chest up, with her gaze slightly averted to the right. ...



world of warcraft Malygos, the blue dragon aspect, cute tee shirt design illustration, 4k

Fig. C.2: Selected Infinity-2B + refiner samples on HPSv3 [44] prompts.

Fig. C.3: Selected Infinity-2B + refiner samples on HPSv3 [44] prompts.

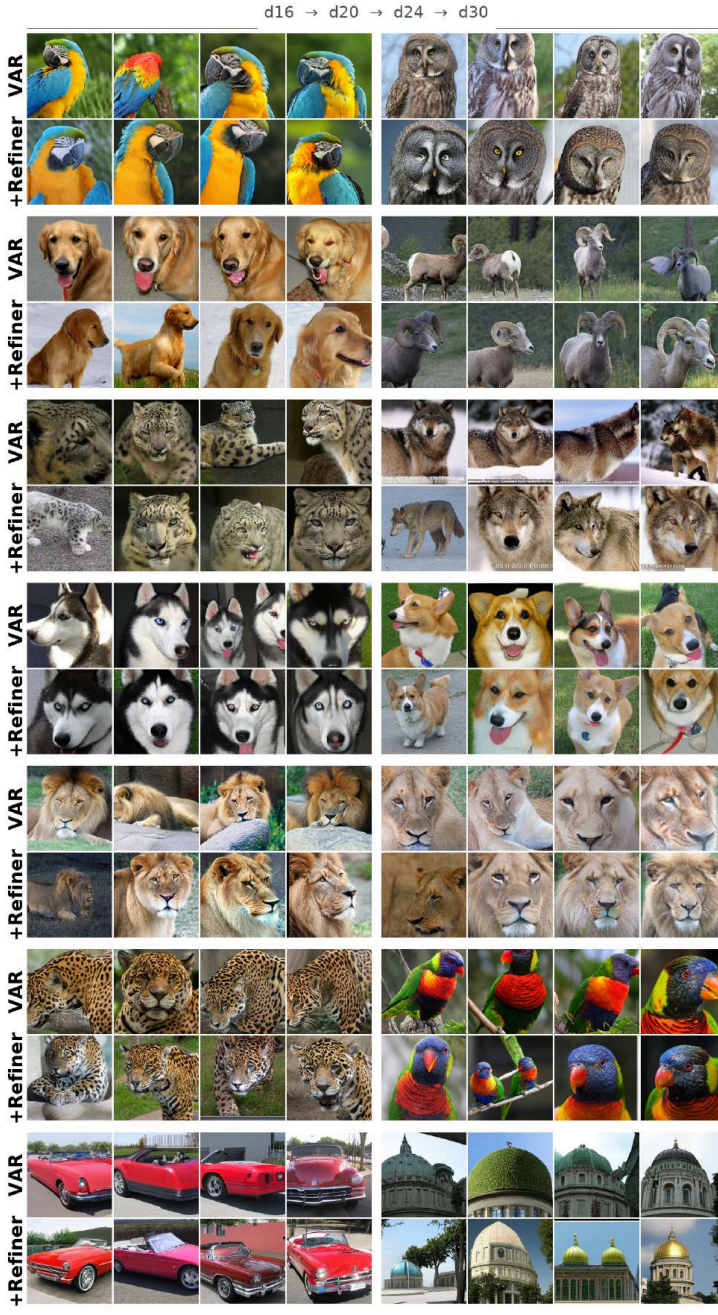
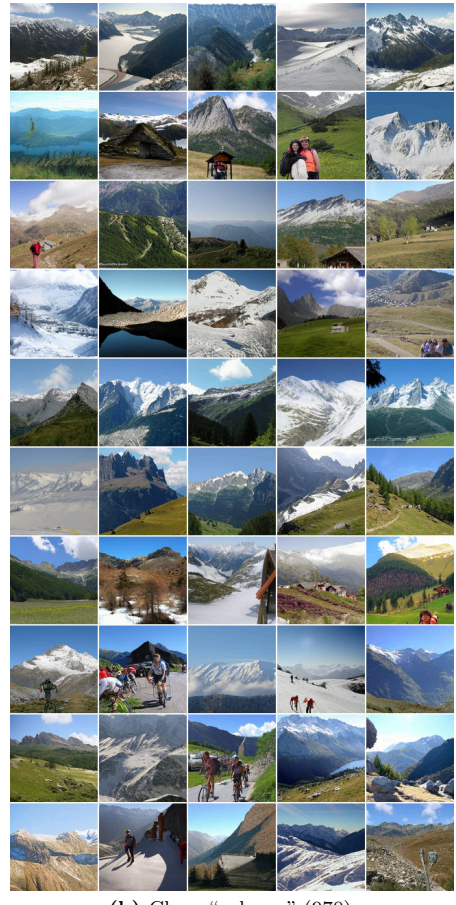


Fig. C.4: Additional Selected Samples from VAR w/o refiner across different scales. We use a classifier-free guidance scale 2.5 with topk=500 for sampling across the scales.

(a) Class: "macaw" (88) **(b)** Class: "Sulphur-crested cockatoo" (89)

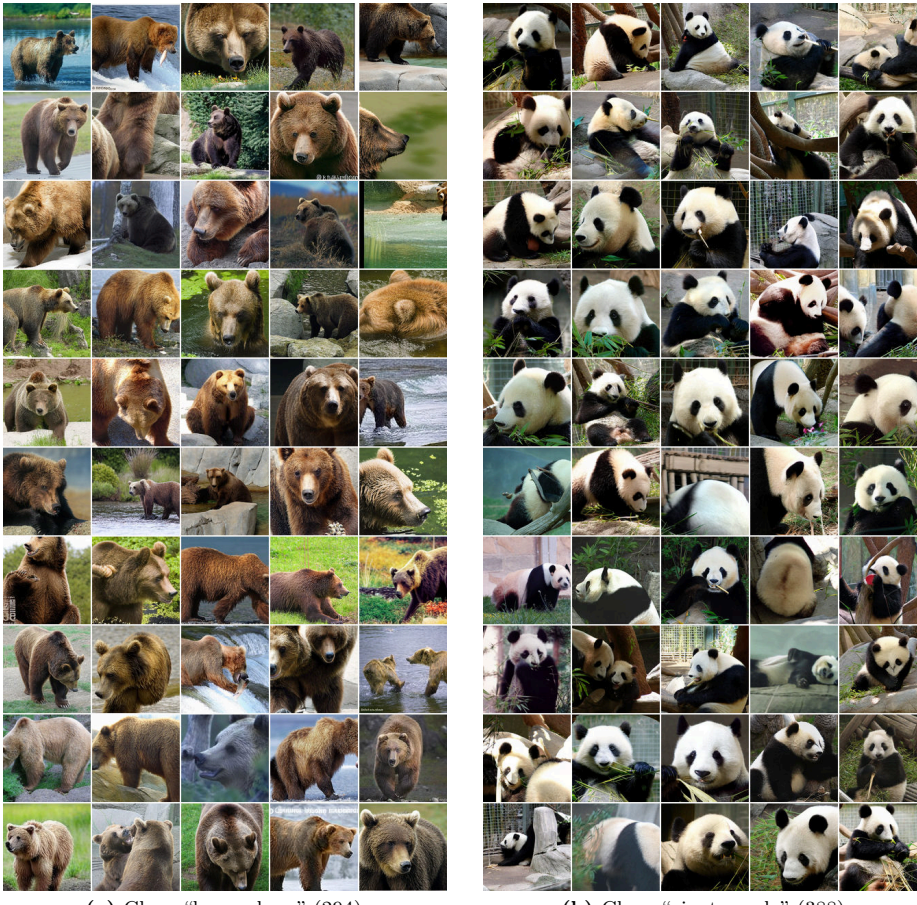
Fig. C.5: Uncurated 256×256 **VARd30 + refiner** samples. Classifier-free guidance scale = 2.5, topk = 500.

(a) Class: “alp” (980)



(b) Class: “volcano” (970)

Fig. C.6: Uncurated 256×256 **VARd30 + refiner** samples. Classifier-free guidance scale = 2.5, topk = 500.



(a) Class: "brown bear" (294) **(b)** Class: "giant panda" (388)

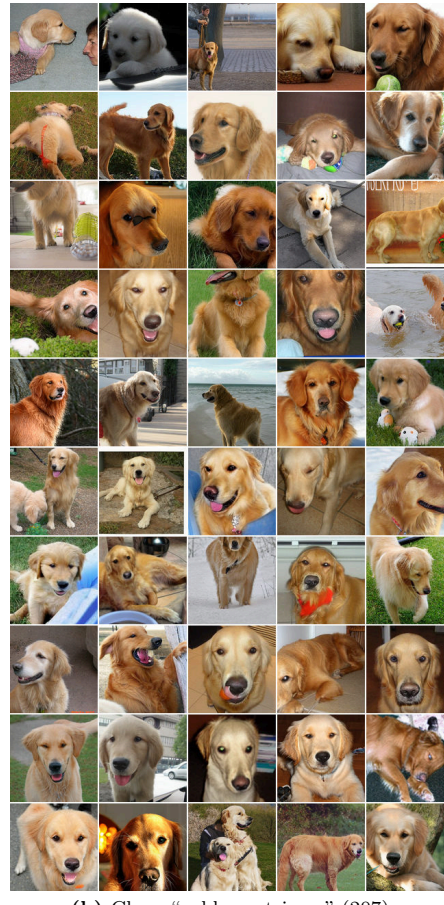
Fig. C.7: Uncurated 256×256 **VARd30 + refiner** samples. Classifier-free guidance scale = 2.5, topk = 500.

(a) Class: “triumphal arch” (873)

(b) Class: “totem pole” (863)

Fig. C.8: Uncurated 256×256 **VARd30 + refiner** samples. Classifier-free guidance scale = 2.5, topk = 500.

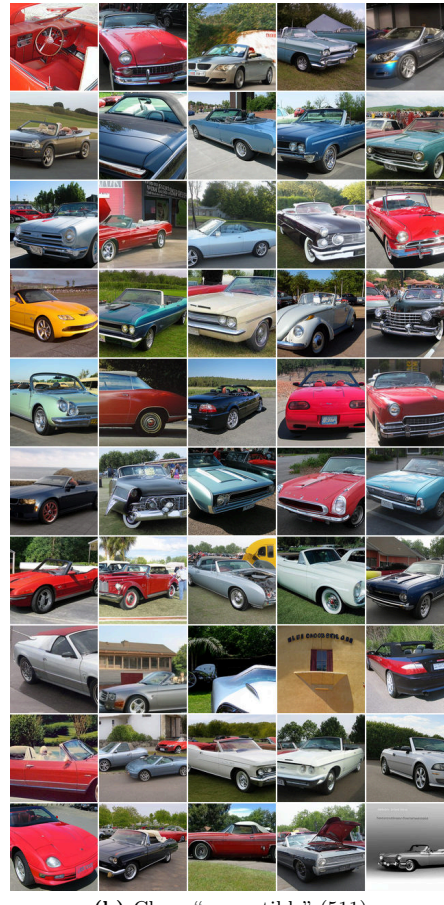
(a) Class: “tabby” (281)



(b) Class: “golden retriever” (207)

Fig. C.9: Uncurated 256×256 **VARd30 + refiner** samples. Classifier-free guidance scale = 2.5, topk = 500.

(a) Class: “electric locomotive” (547)



(b) Class: “convertible” (511)

Fig. C.10: Uncurated 256×256 **VARd30 + refiner** samples. Classifier-free guidance scale = 2.5, topk = 500.