HMAR
Supplementary Material

Contents

A. Extended Related Work

We highlight the key trade-offs between diffusion models, autoregressive image generation, masked generative models, and Visual Auto-Regressive Modeling (VAR), which represents the latest evolution in efficient autoregressive generation. We conclude by discussing efficient attention implementations.

Diffusion Models are the dominant class of generative models for image synthesis. Introduced by [40] and further developed by [19], these models learn to reverse a gradual noising process, enabling high-quality image generation. Subsequent works have improved their efficiency [12, 27, 41], extended them to conditional generation [12], and applied them to various domains including text-to-image synthesis [34, 37]. Diffusion models are preferred over previous image generation methods [16, 21, 35] due to their superior image quality, diversity, and training stability, despite higher computational costs.

Autoregressive Image Generation models offer an alternative approach to image synthesis, drawing inspiration from the remarkable success of next-token prediction in natural language processing [4, 33, 47]. These models generate images sequentially, predicting each new token based on all previous ones, typically following a raster scan pattern. Early works [38, 48, 49] operated directly in pixel space but faced significant computational challenges. More recent approaches have improved efficiency by using Vector Quantized VAEs [14, 22, 50] to compress images into discrete tokens for autoregressive generation, inspiring works like Parti [53] and LlamaGen [42]. While these models benefit from conceptual simplicity and potential transfer learning from language models, they still face challenges in generation speed and quality compared to diffusion models.

Masked Generative Models provide an alternative approach to improve the sampling speed of autoregressive models. These models generate images using a masked prediction objective similar to BERT [3, 11, 18]. By predicting multiple masked tokens in parallel, these models achieve faster generation speeds compared to next-token autoregressive image models. The inherent independence assumption between masked tokens during parallel prediction can lead to inconsistencies or artifacts in the generated images. This approach has been explored in various recent works [6, 7, 23, 52], and a unifying framework for these models is presented in MAR [24], categorizing them as Masked Autoregressive models.

Visual Auto-Regressive Modeling (VAR) [45] enhances the efficiency and quality of autoregressive image generation. Its versatility is demonstrated by successful adaptations for various tasks, including text-to-image generation [17, 28, 51, 54], controllable image generation [26], depth estimation [15], and video generation [20]. Furthermore, VAR has been effectively integrated with other techniques, such as residual diffusion [43] for improved image quality, speculative decoding [8, 44], foldable tokens [25] for enhanced efficiency, solidifying its position as a powerful backbone for autoregressive image generation. However, these models still suffer from quality, efficiency, and flexibility issues.

Efficient Attention Implementations like FlashAttention [9, 10, 39] allow to compute self-attention efficiently on GPU but only support a limited number of attention patterns. Xformer's Memory Efficient Attention [31] supports a wider range of attention patterns but provides memory optimization with limited speedup. Recent work, FlexAttention [30] supports a wider range of attention patterns while providing speedup. However, FlexAttention currently only supports sequence lengths that are multiples of 128, is not optimized for H100 GPUs [30], and finally its flexibility comes at a 10% to 20% performance cost [30].

B. Efficient Attention Computation

We demonstrate how next-scale prediction is adversely affected by longer sequences in comparison to AR models, and how increasing the number of sampling steps results in even longer sequences relative to HMAR. Furthermore, we analyze the attention patterns in VAR and HMAR, highlighting why HMAR performs effectively when conditioned solely on the previous scale. Finally, we present microbenchmarks to evaluate the performance of attention computation using our optimized kernels.

B.1. Long Sequences in Next-Scale Prediction

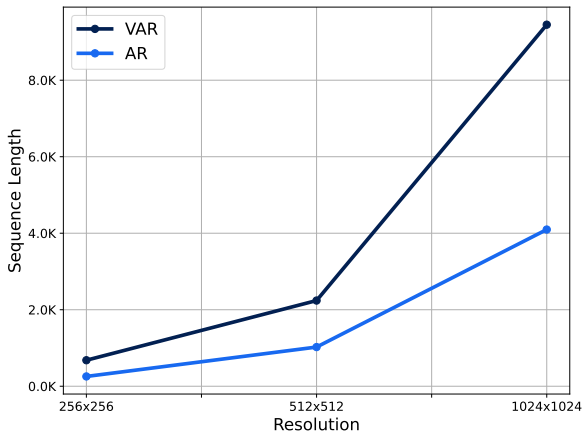


Figure 7. **Sequence Length vs Resolution for next-scale (VAR) and next-token (AR) prediction**. Next-scale prediction requires longer sequences compared to next-token prediction.

Fig. 7 compares the sequence lengths in the next-scale prediction of VAR and next-token autoregressive image generation algorithms like VQ-GAN [14] and LlamaGen[42]. As we grow to higher resolutions, the context length grows making it expensive to train VAR models compared to next-token prediction models. Fig. 5 illustrates the positive impact of increased sampling steps through masking on generation quality. While beneficial, achieving this with (VAR) presents several drawbacks. Each additional sampling step requires a correspondingly longer sequence

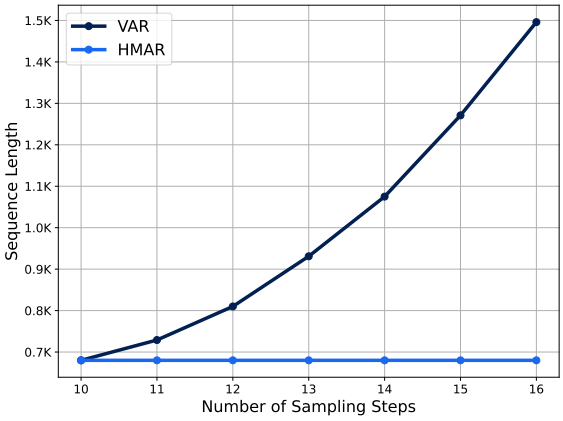


Figure 8. **Impact of Additional Sampling Steps on Sequence Length**: In VAR compared to HMAR, each additional sampling step leads to longer sequence lengths. We show comparisons for 256×256

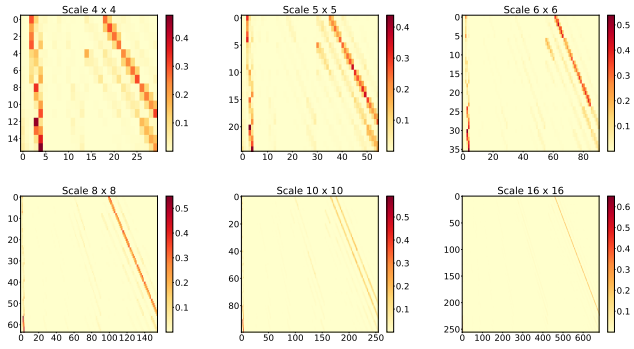
length, as shown in Fig. 8. This increased sequence length has consequences for VAR training, leading to higher computational costs, longer inference times, and greater memory requirements. Our Hierarchical Masked Autoregressive (HMAR) formulation, in contrast, allows for a flexible increase in the number of sampling steps without necessitating any changes to the sequence length. Furthermore, VAR models are inherently limited in their maximum number of sampling steps by the number of available scales. As a concrete example, a 16×16 latent space restricts VAR to a maximum of 16 sampling steps. HMAR overcomes this limitation, enabling up to 256 sampling steps without requiring the re-masking of previously unmasked tokens. If re-masking is allowed, HMAR can theoretically accommodate an arbitrary number of sampling steps.

B.2. Attention Pattern Analysis

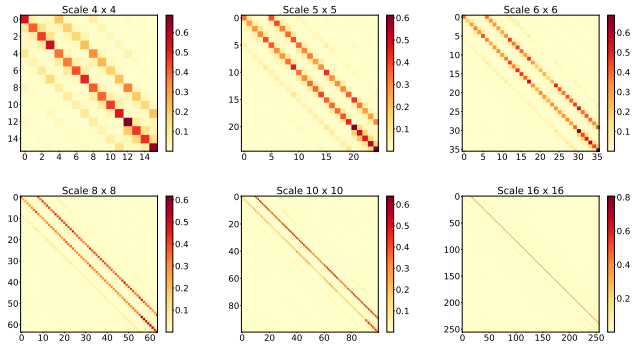
The attention patterns visualized in Fig. 9 reveal that tokens primarily attend to their local neighborhoods. This localized attention behavior provides strong empirical support for our hypothesis regarding next-scale prediction, demonstrating that the most relevant information for predicting the next scale is predominantly contained in the immediately preceding resolution level. This finding led us to streamline our model by replacing the full prefix conditioning mechanism with a simpler approach, where each scale only depends on its direct predecessor, notably preserving the model's predictive performance while achieving significant computational efficiency gains.

B.3. Attention Patterns

We provide an illustration of the attention masks utilized in VAR and HMAR in Fig. 10 to highlight their distinct mechanisms. VAR employs a block-causal attention mask, which allows each scale to attend not only to itself but also to all preceding scales. This design ensures a comprehensive flow of information across scales, facilitating a more global understanding of the data. In



(a) Normalized Attention Scores VAR



(b) Normalized Attention Scores in our HMAR

Figure 9. **Comparing Attention Patterns in VAR and HMAR**

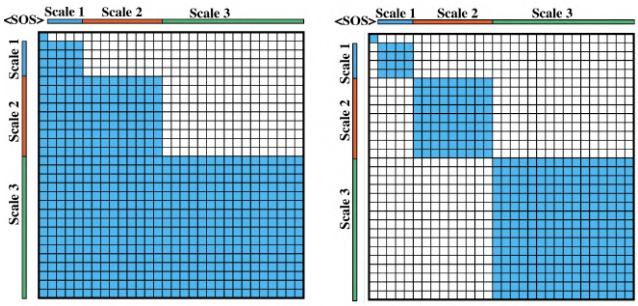


Figure 10. **Attention masks in VAR and HMAR**: Block-diagonal pattern in HMAR (right) enables more sparsity compared to the block-causal pattern in VAR(left).

contrast, HMAR adopts a block-diagonal attention mask, where each scale is restricted to attending only to the immediately preceding scale in order to generate the next one. This results in a more localized and computationally efficient attention mechanism. As shown in Fig. 10, the block-diagonal mask is significantly sparser compared to the block-causal mask. This sparsity can be leveraged to achieve faster attention computation, particularly as the degree of sparsity increases with image resolution. Consequently, this approach becomes even more efficient for attention computation at higher resolutions.

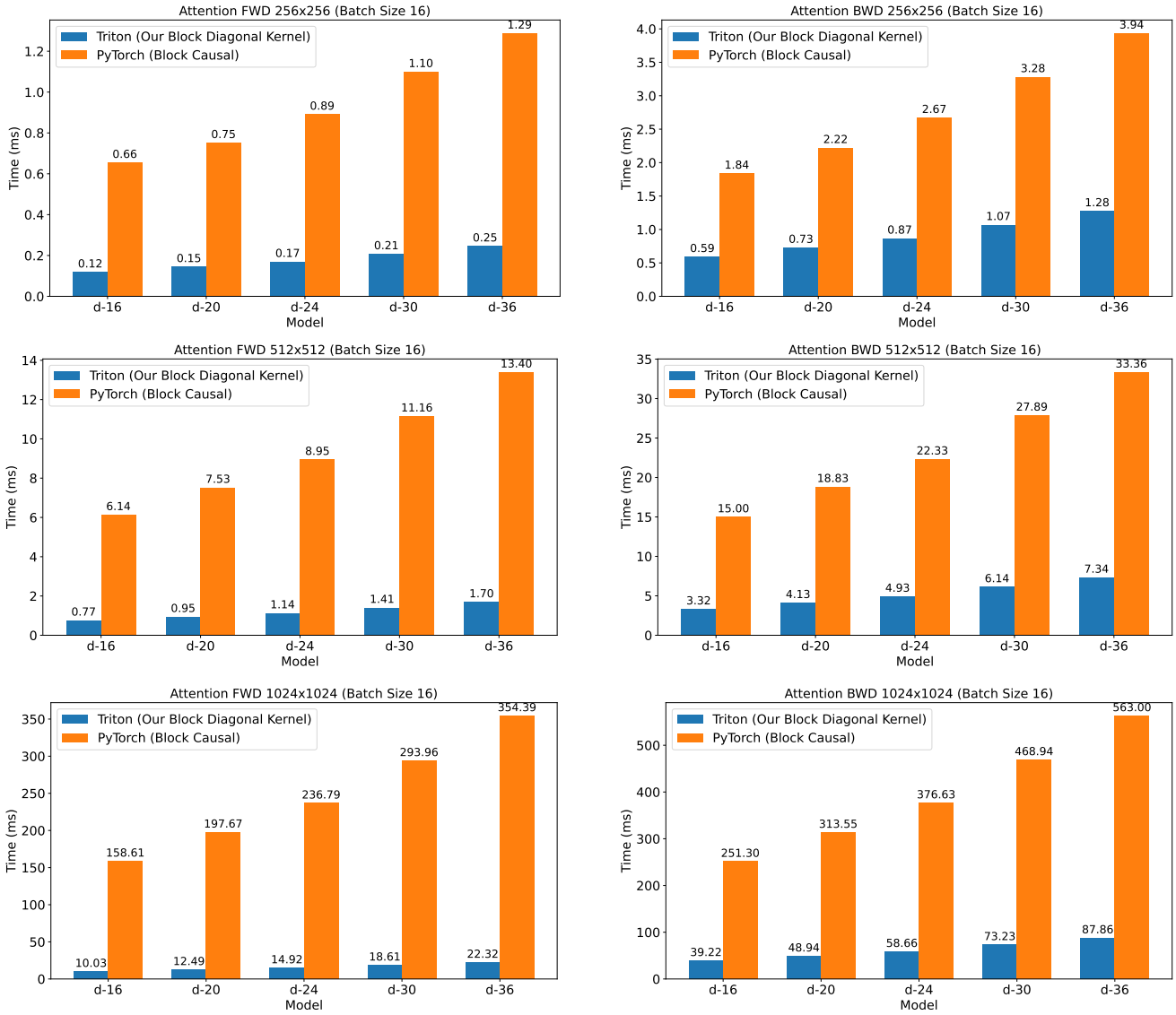


Figure 11. **Comparison of forward and backward pass speeds between our block sparse attention kernel in (HMAR) and the PyTorch block-causal attention in (VAR)** across different image resolutions (256×256, 512×512, and 1024×1024). Tests were performed on an A100 80GB GPU with batch size 16 and model dimension 64. Our implementation shows significant speedups, achieving up to 15.8× faster forward pass and 6.4× faster backward pass at 1024×1024 resolution.

B.4. Efficient Attention Performance

We benchmark the performance of our block sparse attention kernel used in HMAR against the block-causal attention in VAR. For the block-causal attention, we compare against the memory-efficient attention implementation that supports different attention masks via `torch.sdpa`. The benchmarking is conducted across various image resolutions, utilizing a single A100 80GB GPU and results are averaged over 25 repetitions. Results are shown in Fig. 11. At each resolution, we evaluate both the forward and backward passes, ensuring a comprehensive analysis of performance. The benchmarks are conducted with a batch size (bs) of 16 and a model dimension (d) of 64. For each corresponding model, such as d-24 or d-20, the number of attention heads matches the model dimension—for instance, the d-24 model has 24 attention heads.

On the forward pass, our efficient implementation is up to $5.2\times$ faster at 256×256, $7.9\times$ faster at 512×512, and $15.8\times$ faster at 1024×1024. On the backward pass, our efficient implementation is up to $3.1\times$ faster at 256×256, $4.6\times$ faster at 512×512, and $6.4\times$ faster at 1024×1024, demonstrating significant performance improvements of our implementation across various resolutions.

C. Training Dynamics

In this section, we delve into how each scale contributes to visual quality and how to focus the model's capacity during training on the most important scales that matter for visual quality.

C.1. Learning Difficulty Across Scales

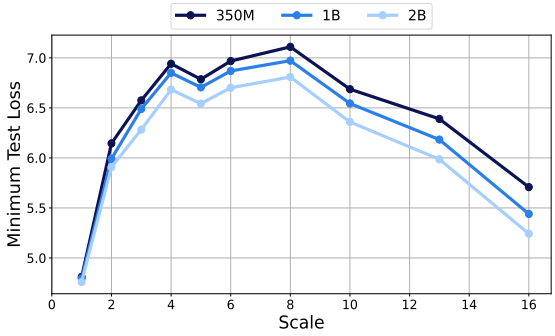


Figure 12. **Minimum Test Loss Across Scales**

We use the minimum test loss at each scale as a proxy for learning difficulty. We find that this has an approximately log-normal pattern (Fig. 12), suggesting that scales in the middle are more challenging to learn compared to those at the beginning and end.

C.2. Loss Weighting Ablation

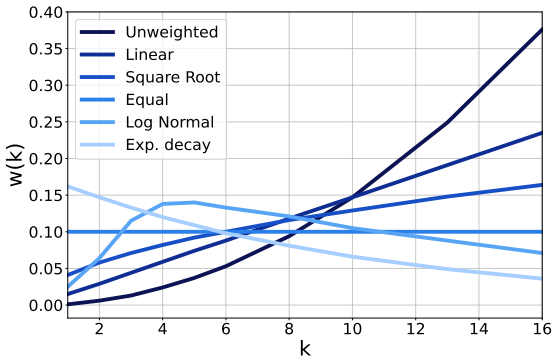


Figure 13. **Different Loss Weighting Functions**

We ablate the impact of loss weighting on image quality. For each function (Fig. 13), we train a 310M parameter model for approximately 150K steps and evaluate the FID, Inception Score, Precision, and Recall. Results in Table. 4 demonstrate that loss weighting can significantly influence the quality, with the log-normal weighting yielding the best performance.

C.3. Loss Analysis

Unlike autoregressive language models, where total loss correlates with downstream performance, we find this relationship doesn't hold in our setting. Due to the disproportionate number of tokens in later scales, the total loss is heavily influenced by performance on high-frequency details that are often imperceptible to human observers. As shown in Fig. 16, models with better performance in early and middle scales achieve superior FID and Inception Scores, despite potentially higher total losses.

Loss Weighting Ablation on ImageNet-256×256				
Function	**FID ↓**	**IS ↑**	**Prec ↑**	**Rec ↑**
Unweighted	3.89	283.3	0.86	0.48
Equal	3.64	296.5	0.85	0.50
Linear	3.72	301.9	**0.86**	0.50
Sqrt.	3.79	306.6	0.86	0.50
Exp. decay	3.72	281.1	0.84	0.50
Log-normal	**3.59**	**307.4**	0.85	**0.50**

Table 4. **Loss Reweighting Ablation on ImageNet-256 × 256.** (cfg=1.5, top-k=900, top-p=0.96) We show the impact of the choice of loss weighting on image quality. A log-normal weighting that mirrors the distribution of learning difficulty yields the best performance.

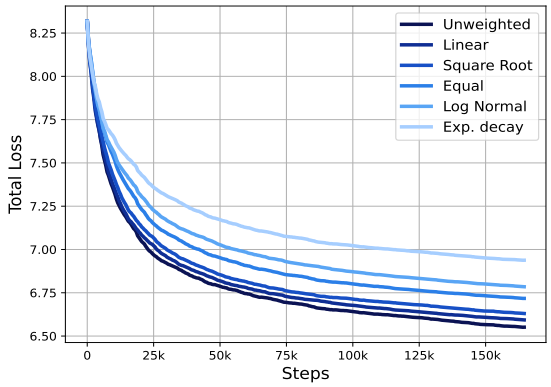


Figure 14. **Total Loss for Different Weighting Functions**

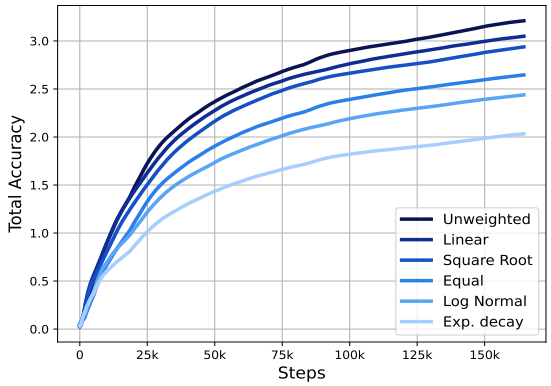


Figure 15. **Total Accuracy for Different Weighting Functions**

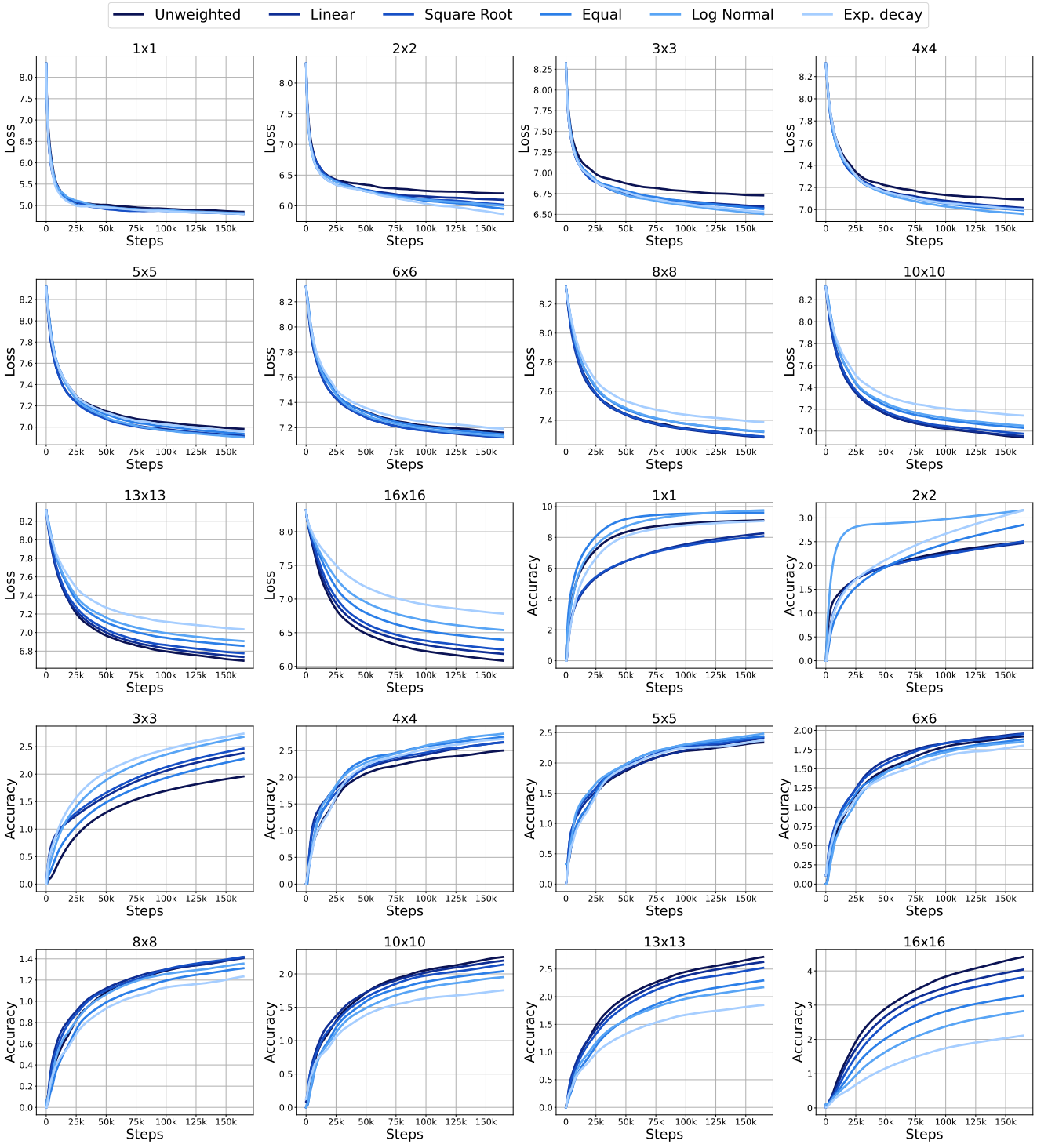


Figure 16. **Scale-wise Decomposition of Loss and Accuracy**. We analyze the impact of loss weighting across scales on generated image quality. By decomposing both loss and accuracy metrics scale-by-scale, we reveal a key insight: prioritizing performance in early and intermediate scales, rather than simply minimizing total loss, leads to improved perceptual quality. This is evidenced by the correlation between strong early/mid-scale performance and superior FID and Inception Scores, as detailed in Table 4. This suggests that while later scales contribute significantly to the overall loss due to their higher token count, they are less critical for generating high-fidelity images.

D. Masking/ Parallel Sampling

In this section, we start by demonstrating how parallel sampling in VAR can result in error accumulation during the image generation process. We then delve into our masked image reconstruction objective and discuss masked sampling. Finally, we present ablations to illustrate how intra-scale masking improves the generation quality.

D.1. Error Accumulation in Parallel Sampling

Figure 17. **Error Accumulation in Next-Scale Prediction**: Teacher forcing experiment showing error propagation across scales. The top rows show image generation starting from the ground truth at earlier scales, showing lower quality. The bottom rows show image generation starting from later scales, leading to improved quality, indicating error accumulation from earlier scales impacts quality. Model: VAR-d16.

We illustrate error propagation in image generation using a VAR-d16 model through a teacher-forcing experiment. Ground truth tokens are provided at each scale, and the model generates the rest of the image beginning from that scale. As shown in Fig. 17, starting from earlier scales results in poorer quality, while later scales yield better images. This suggests that to get good image quality, it is important to get the earlier scales correctly. We hypothesize that errors introduced during parallel token sampling at earlier scales propagate during the generation process and impact the overall image quality.

D.2. Image Reconstruction

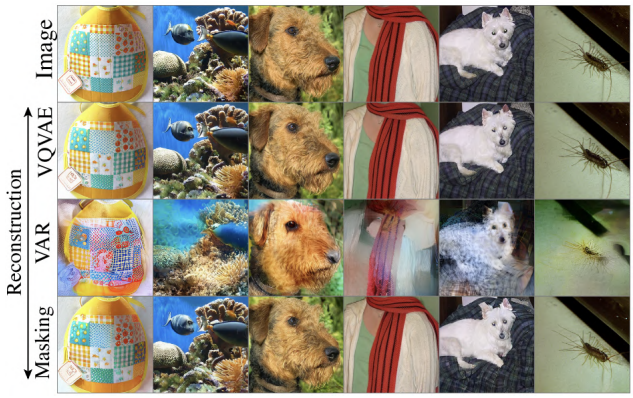


Figure 18. **Image Reconstruction** in VQ-VAE, VAR and HMAR

Fig. 18 compares image generation in HMAR and VAR. Our HMAR model can reconstruct images to a quality comparable to the Multi-Scale VQ-VAE. We mask out 50% of all levels in the image and use HMAR to predict the masked tokens, then combine these tokens across all scales to form a complete image. In contrast, we also demonstrate VAR's image reconstruction. For each image, we provide the ground truth at each level, as done during VAR training, and allow the VAR model to predict the next scale.

D.3. Masked Finetuning

We find masked prediction significantly easier to learn than next-scale prediction, reaching accuracies of 65+% compared to just 5% in next-scale prediction (Fig. 22), allowing for efficient fine-tuning of a masked prediction layer atop a pre-trained next-scale model. Unlike in next-scale prediction, where we reweight the loss by each scale, we find that for masked prediction, the task is easier to learn, and as such, reweighting the loss to prioritize certain scales can actually hurt performance. We use 25% to 40% of the pre-training iterations when fine-tuning. In our experiments, we fine-tune specific layers while freezing others, but alternative PEFT methods like LORA could also be used effectively to reduce the number of parameters used. Directly pre-training with both next-scale and masked-prediction objectives is another exciting direction to explore.

Figure 19. **Multi-Scale VQ-VAE [45] Reconstructions**: We show some failure cases for the Multi-Scale VQ-VAE [45]. While it can capture general image structures, it struggles to reconstruct fine-grained details, particularly in complex elements such as text and facial features.

D.4. Masked Sampling

We run a hyperparameter search to choose the sampling steps and report quantitative metrics in Table 1 with 14 steps because this yields the best performance. In particular, additional sampling steps at the earlier scales have the highest impact on quantitative metrics. We find that additional sampling steps beyond the first five scales do not improve quantitative metrics but can improve finer details in the image (Fig. 5). We show how adding one additional sampling step to each scale in HMAR-d16 impacts the FID in Fig. 20.

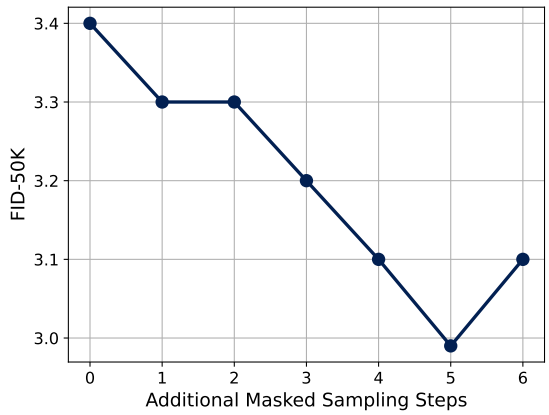


Figure 20. **Impact of Extra Masked Sampling Steps on FID for HMAR-*d*16.** Increasing the number of sampling steps at earlier scales leads to better FID.

We use classifier-free guidance when sampling and show its effect on FID and Inception score in Fig. 21.

E. Discussion on Multi-Scale VQ-VAE Tokenizer

We adopt the Multi-scale VQ-VAE tokenizer from VAR [45]. In this section, we provide more details and highlight some of its limitations and possible ways to improve it.

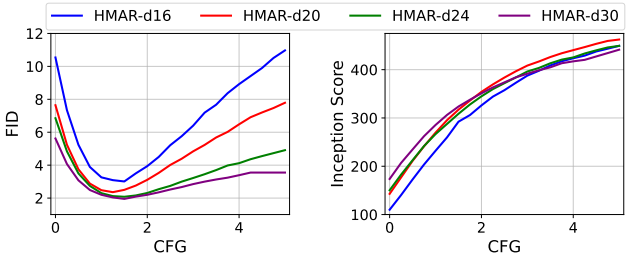


Figure 21. **Classifier-free guidance** impact on FID and Inception Score

E.1. Image Reconstruction

We provide PSNR and rFID values for the tokenizer at different image resolutions in Table 5. In Fig. 19, we show the tokenizer's

Resolution	rFID	PSNR (dB)
256×256	0.92	20.69
512×512	0.66	21.74

Table 5. **PSNR and rFID** values for different image resolutions

inability to capture fine-grained details within images. Recent works like HART [43] propose a hybrid tokenization scheme, incorporating continuous tokens to better represent high-frequency details. These contributions are complementary to our work.

E.2. Codebook Utilization

In Fig. 23, we analyze the distribution of codebook usage across different scales. The overall codebook usage appears relatively uniform; however, early scales, which capture coarse structural features, exhibit highly skewed distributions, with only a small subset of codebook entries being used. In contrast, later scales that capture fine-grained details demonstrate a more uniform distribution of code usage. Future tokenizer designs could benefit from an asymmetric approach: smaller, specialized codebooks for early scales to efficiently capture essential structural features and larger, more diverse codebooks for later scales to accommodate the broader range of local details.

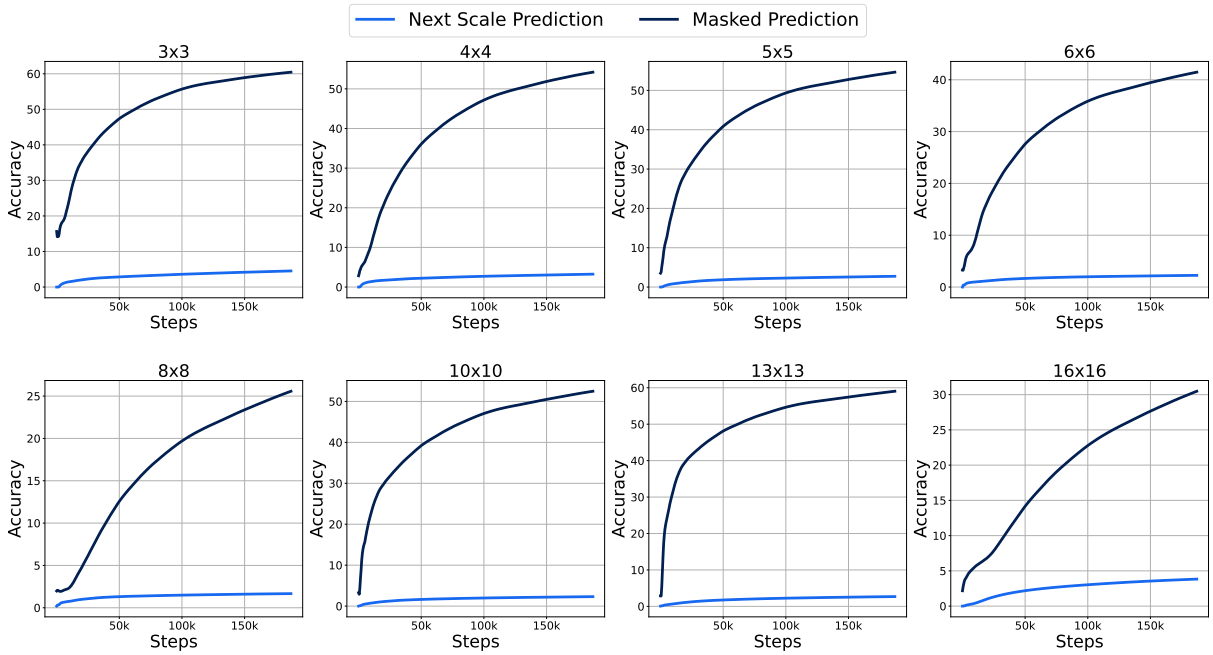


Figure 22. **Comparison of Accuracy Between Next-Scale Prediction and Masked Prediction Across Scales**. Masked prediction on residuals is a considerably simpler task to learn compared to next-scale prediction.

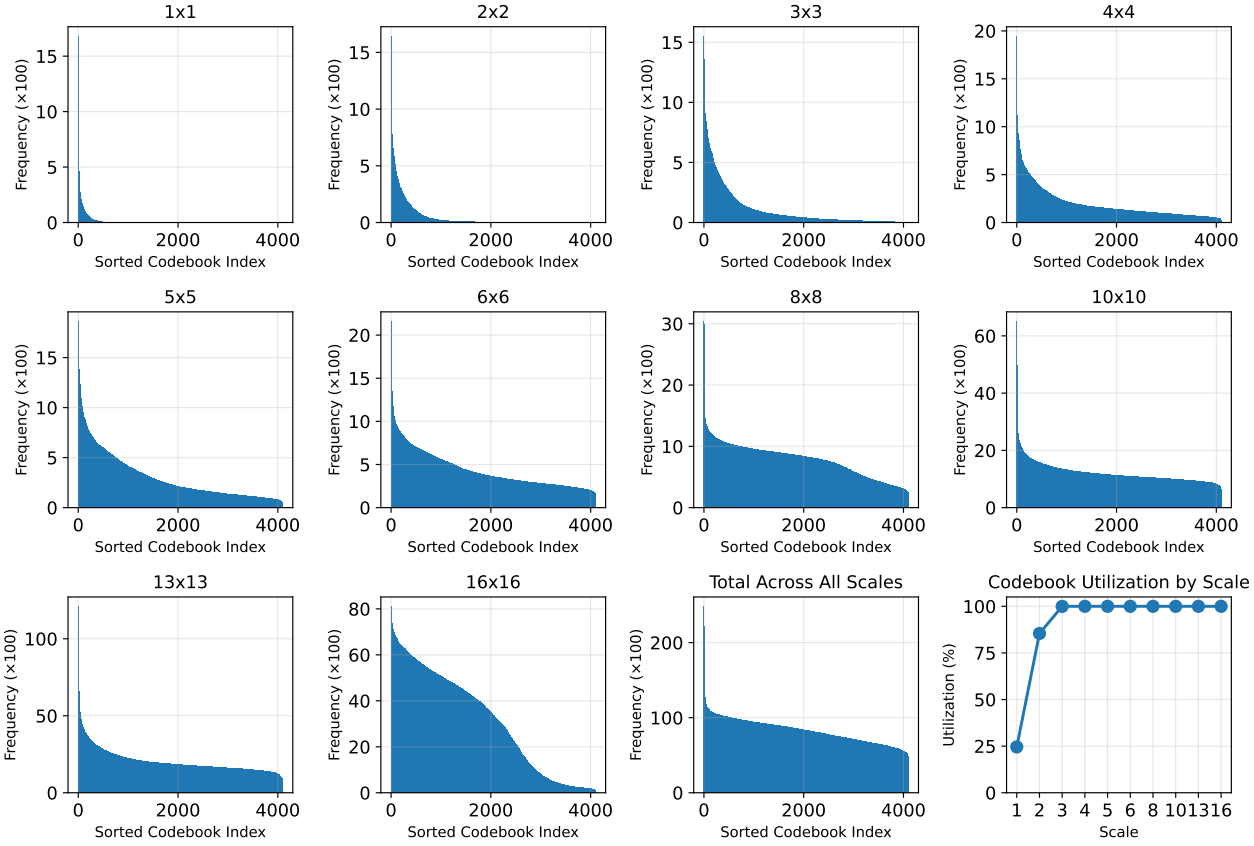


Figure 23. **Analysis of Codebook Usage Patterns Across Scales**. We observe distinct patterns in how codebook entries are utilized: early scales show highly skewed distributions with only a small subset of codes being frequently accessed, indicating potential redundancy, while later scales demonstrate more uniform usage patterns. The codebook utilization rate progressively increases from less than 50% in early scales to nearly 100% by scale 4.

F. Additional Qualitative Results

F.1. Qualitative Comparisons

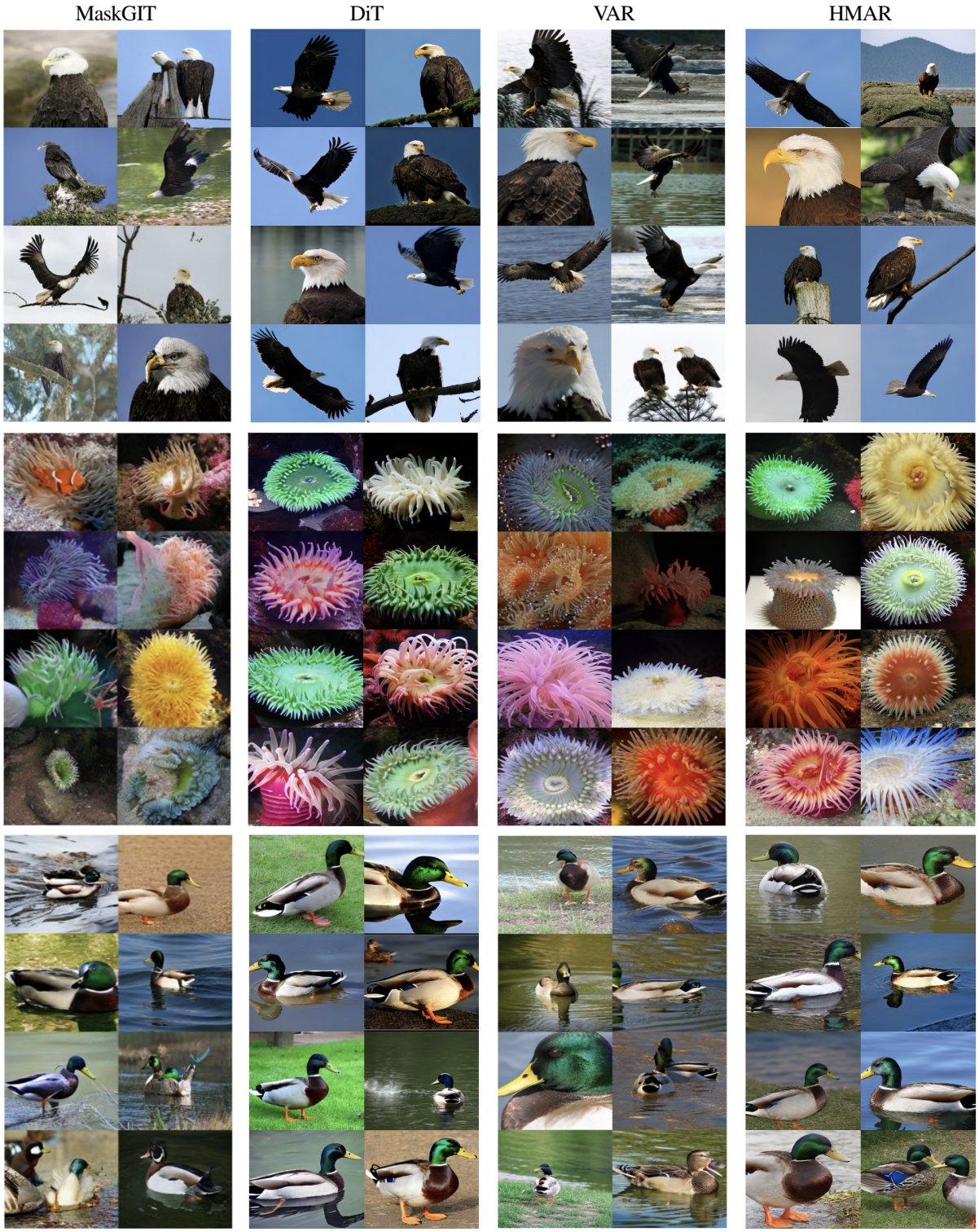


Figure 24. **Qualitative Comparisons on ImageNet** 256×256

F.2. Class Conditional ImageNet 256x256 Samples

Class ID 162, Beagle

Class ID 145, Penguin

Class ID 417, Balloon

Class ID 437, Beacon

Class ID 975, Lakeside

Class ID 285, Egyptian Cat

Figure 25. **Additional Class-Conditional Image Generation Samples on ImageNet** 256×256

Class ID 323, Monarch

Class ID 350, Ibex

Class ID 269, Timber Wolf

Class ID 984, Rapeseed

Class ID 148, Killer Whale

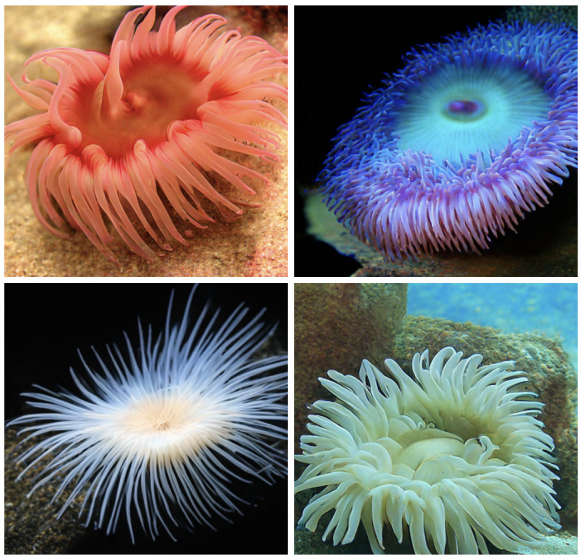
Class ID 296, Ice Bear

Figure 26. **Additional Class-Conditional Image Generation Samples on ImageNet** 256×256

F.3. Class Conditional ImageNet 512x512 Samples

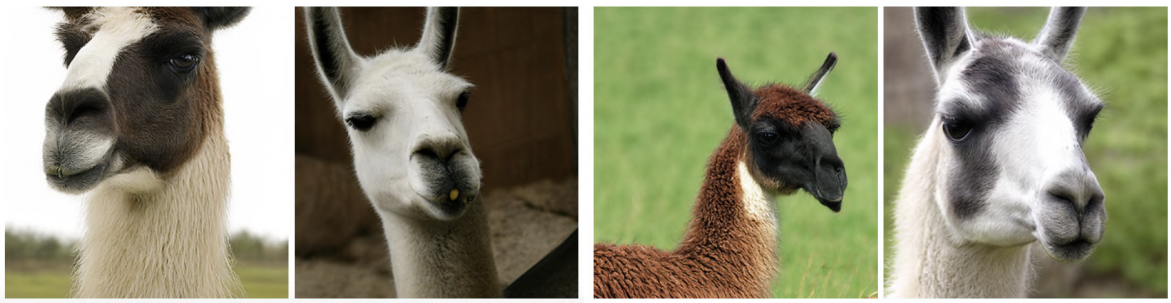
Class ID 3, Shark

Class ID 22, Bald Eagle



Class ID 108, Sea Anemone

Class ID 388, Giant Panda



Class ID 355, Llama

Figure 27. **Additional Class-Conditional Image Generation Samples on ImageNet** 512×512