EVAR: Edge Visual Autoregressive Models via Principled Pruning

Zefang Wang[1,2*], Ying Li[2], Yanyu Li[3], Mingluo Su[2], Simin Xu[2], Guanzhong Tian[1†], and Huan Wang[2†]

[1] Zhejiang University, China
[2] Westlake University, China
[3] Snap Inc., USA

Project Page: https://aden9460.github.io/EVAR

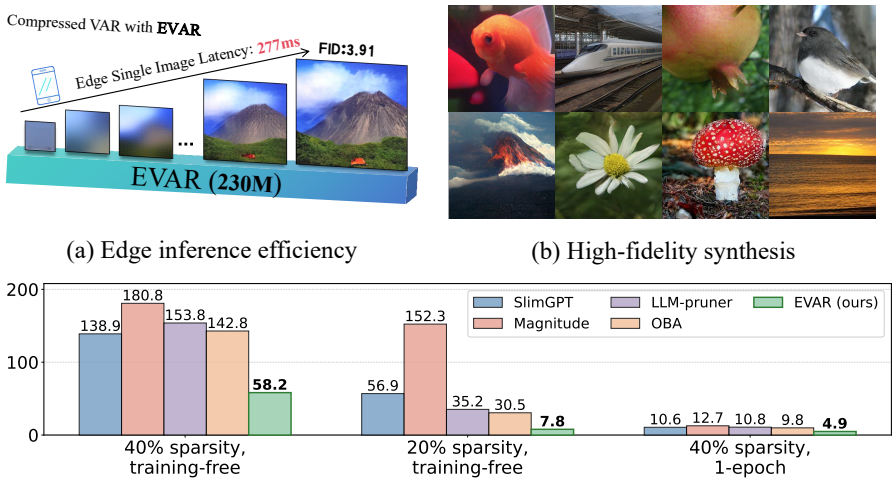


Fig. 1: Generative performance and on-device deployment of EVAR. (a) Edge inference efficiency: EVAR achieves a single-image latency of 277 ms and a competitive 3.91 FID on iOS. (b) High-fidelity synthesis: Qualitative samples generated by our pruned 230M model on ImageNet 256 × 256. (c) Pruning comparison: EVAR achieves better FID than state-of-the-art pruning baselines across different pruning settings.

Abstract. Next-scale visual autoregressive (VAR) models offer strong generative fidelity but remain computationally prohibitive for resource-constrained edge devices. We introduce EVAR, a structured-pruning framework tailored to next-scale VAR models. We find that standard pruning paradigms are poorly matched to multi-scale VAR architectures: rapid token growth at later scales dominates Hessian accumu-

† Corresponding to: Huan Wang ⟨wanghuan@westlake.edu.cn⟩, Guanzhong Tian ⟨gztian@zju.edu.cn⟩

⋆ Work done while Zefang was a visiting intern at ENCODE Lab, Westlake University.

lation, marginalizing critical coarse scales and amplifying autoregressive error cascading. To address this issue, EVAR introduces Pyramid-Aware Optimal Brain Surgeon. Through pyramid-aware Hessian accumulation, Pyramid-Aware OBS rebalances scale-wise activation statistics and biases the solver toward preserving coarse-scale representations while maintaining an invertible Hessian estimate for closed-form weight compensation. Additionally, we propose Progressive Scale-Aware Distillation (PSAD) to counteract scale-wise gradient imbalance during fine-tuning. On ImageNet benchmarks, EVAR substantially reduces parameters and model footprint while retaining competitive quality. On an iOS deployment, EVAR further cuts single-image latency from 494 ms to 277 ms **(1.8× speedup)**, with FID changing only marginally.

Keywords: Visual Autoregressive Models · Structured Pruning · Edge Deployment

1 Introduction

Autoregressive (AR) [9, 23, 29, 42, 45, 52] models have recently made remarkable progress in visual generation, with strong scalability enabling high-quality image synthesis and editing. However, conventional AR models rely on a next-token prediction paradigm and generate tokens strictly sequentially, so the token-by-token decoding requires numerous steps and leads to significant inference latency.

Next-scale visual autoregressive (VAR) modeling [7, 8, 16, 25, 26, 28, 32, 37, 43, 44, 51, 53–55] shifts this paradigm from next-token to next-scale prediction. Instead of emitting tokens one by one, VAR decodes visual content in a coarse-to-fine hierarchy, generating all tokens in parallel within each scale. This design reduces the number of decoding steps while retaining the expressive power of AR models, enabling performance comparable to state-of-the-art diffusion models on image synthesis, super-resolution, and inpainting. Yet, these models remain large and computationally demanding. Furthermore, recent throughput-oriented VAR acceleration methods obtain stronger speedups under large-batch settings, leaving latency-critical single-image inference on edge devices less explored [8]. This limitation introduces our key challenge: enabling efficient, low-latency single-image generation on resource-constrained edge hardware [4, 27].

These constraints necessitate efficient model compression techniques [5, 6, 10, 11, 19, 39, 49, 50, 57] that reduce the computational footprint. Among them, OBS-style second-order pruning [18, 20] is particularly attractive because it estimates pruning-induced output perturbations and provides closed-form weight compensation. However, directly applying existing OBS-style structured pruning methods to next-scale VAR models is suboptimal, because these methods are mainly designed for LLMs and do not explicitly account for VAR's multi-scale generation mechanism. As illustrated in Fig. 2, the multi-scale nature of VAR introduces a critical vulnerability: scale-wise token imbalance coupled with error cascading. Coarse-scale tokens dictate the global structural layout of the image (Fig. 2 a), yet they account for a minute fraction of the total sequence length. If

pruning introduces errors at these early stages, the distortions can accumulate and be progressively amplified throughout the autoregressive pipeline (Fig. 2 b). Consequently, directly applying OBS-style Hessian-based pruning to VAR leads to biased Hessian estimation: flattened token accumulation is dominated by abundant fine-scale tokens, which underrepresents the critical coarse scales that control global semantics.

To resolve this architectural mismatch, we propose *EVAR*, an efficient structured-pruning framework tailored to next-scale VAR models and edge deployment. At the core of EVAR lies Pyramid-Aware OBS. Instead of flat token concatenation, we reconstruct the Hessian accumulation with token-count normalization and scale-aware decay weighting. This formulation biases the second-order solver toward preserving pruning-sensitive coarse scales, which are critical for global image structure. Simultaneously, streaming calibration over the full multi-scale sequence, together with a damping term, yields a positive-definite Hessian estimate required for analytical closed-form OBS compensation. As a result, EVAR makes the pruning process more principled for the multi-scale structure of VAR.

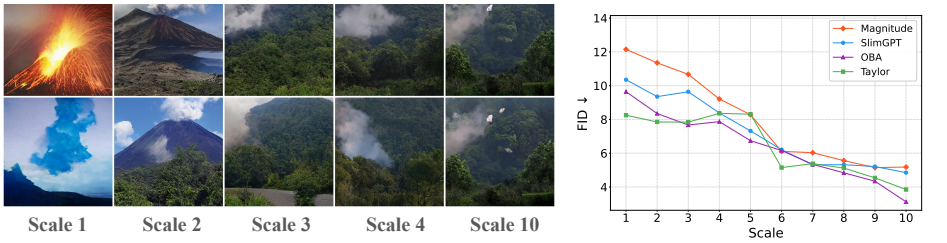
Beyond pruning stage, we observe that scale-wise imbalance also affects the post-pruning fine-tuning phase. Standard training objectives are dominated by abundant fine-scale tokens, thereby over-emphasizing local details while under-supervising coarse-scale semantics. EVAR addresses this with a Progressive Scale-Aware Distillation (PSAD) scheme. By combining a coarse-to-fine distillation curriculum with scale-aware loss weighting, PSAD balances gradient contributions across the generative hierarchy and improves post-pruning recovery.

To evaluate EVAR under realistic conditions, we build an on-device execution pipeline via Core ML. Rather than relying purely on theoretical MACs, we directly measure the on-device latency under different compute-unit settings. EVAR delivers competitive generative performance while unlocking efficient mobile and edge deployment of visual autoregressive models.

To the best of our knowledge, EVAR is the **first** OBS-based structured pruning framework explicitly designed for next-scale visual autoregressive models. Furthermore, we deploy the compressed model on iOS devices, enabling efficient and high-fidelity on-device image generation.

Our contributions can be summarized as follows:

- We identify an architectural mismatch between standard pruning paradigms and next-scale VAR generation, showing that scale-wise token imbalance and autoregressive error cascading can compromise global structure and semantics.
- We introduce *EVAR*, featuring *Pyramid-Aware OBS*, a novel structured pruning framework that utilizes Pyramid-Aware Hessian Accumulation to protect global semantics while ensuring matrix positive-definiteness.
- We propose Progressive Scale-Aware Distillation (PSAD) to mitigate scale-wise gradient dominance during fine-tuning, balancing gradient contributions across scales and improving post-pruning recovery.
- We build a Core ML-based iOS deployment pipeline and demonstrate practical on-device generation: EVAR compresses VAR-d16 to 230M parameters,



(a) Visualizing the generative hierarchy (b) Scale-wise sensitivity to pruning

Fig. 2: Scale-wise token imbalance and error cascading in next-scale VAR. **(a)** By isolating the sampling results at specific scales, we observe that coarse-scale tokens (e.g., Scales 1–3) fundamentally dictate the global semantic layout and structural appearance of the image, whereas fine-scale tokens (e.g., Scale 10) predominantly refine local textures. **(b)** We replace individual scales of a dense VAR-d16 model with a 20% sparse model pruned via different methods. Pruning the early scales yields drastically higher FID scores (worse quality), revealing a critical vulnerability: compression errors in coarse scales trigger an *error cascading* effect that continuously accumulates through the autoregressive pipeline, irreversibly distorting the final output.

achieving 277 ms single-image latency and a 1.8× speedup with FID changing only marginally.

2 Related Work

2.1 Efficient Autoregressive Image Generation

Autoregressive (AR) models decompose the joint distribution of an image into a product of conditional distributions and have been a long-standing backbone of generative modeling. PixelRNN and PixelCNN [46, 47] generate pixels sequentially with recurrent or convolutional architectures, but suffer from slow sampling. Inspired by Transformer-based language models [2, 35, 36, 48], large-scale GPT-style decoders have been adopted for image generation, enabling global context modeling and strong scalability. Recent systems, including VQGAN and RQ-Transformer [9, 21], diffusion–AR hybrids [14, 56], and GPT-like image generators such as LlamaGen and Lumina-mGPT [31, 42] further push fidelity by operating in discrete token spaces with powerful next-token prediction.

To reduce the sequential bottleneck of next-token image generation, a line of work studies partially parallel or masked decoding, such as MaskGIT and MAR [3,24]. CoDe [8] improves VAR efficiency with minimal quality loss by using a collaborative decoding strategy, where a large drafter handles small-scale low-frequency structure and a small refiner predicts large-scale high-frequency details. Related parallelized or scalable AR models, such as PAR [52] and AiM [23], further explore efficient image generation with reduced sequential dependency. However, these efforts mainly target generation fidelity, decoding efficiency, or

scalability on powerful GPUs. Model compression and practical deployment of next-scale VAR models on resource-constrained edge devices remain largely underexplored, which is the focus of EVAR.

2.2 Network Pruning

Network pruning [17, 34] is a standard approach to reducing the computational and memory footprint of a pre-trained network by removing or sparsifying redundant parameters. According to pruning granularity, existing methods are typically divided into *unstructured* and *structured* pruning. Unstructured pruning [12,13,18,20,38,40] zeros out individual weights according to saliency scores, enabling fine-grained sparsification at the level of each matrix entry. This granularity often yields smaller performance degradation at a given sparsity level than structured pruning. However, the resulting irregular sparsity is difficult to exploit on general-purpose hardware due to poor alignment with SIMD/SIMT execution, irregular memory access, and limited sparse-kernel support. Consequently, end-to-end speedups are often modest despite high nominal sparsity. Structured pruning [1, 10, 19, 22, 30, 33, 41] removes entire channels or heads, yielding dense, smaller subnetworks that better align with standard GEMM kernels and thus more directly translate into wall-clock acceleration.

3 Method

3.1 Preliminaries on VAR

Visual Autoregressive (VAR) models [44] reformulate autoregressive image modeling by shifting from traditional next-token prediction to a next-scale prediction scheme. Instead of generating an image token by token, VAR represents the image feature map $f \in \mathbb{R}^{h \times w \times C}$ as K token maps $(r_1, r_2, \ldots, r_K)$ at multiple resolutions, where the spatial resolution increases with the scale index and the final token map r_K matches the original feature-map resolution.

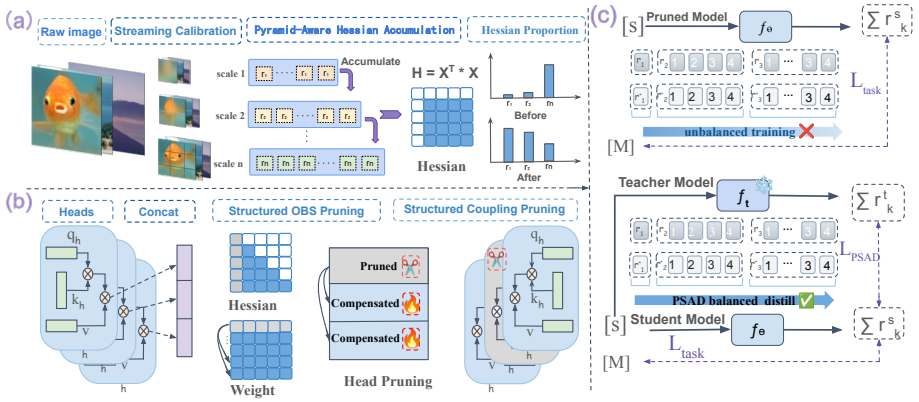
The joint distribution over the multi-scale token maps is factorized as

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

where $r_k \in [V]^{h_k \times w_k}$ denotes the token map at scale k, and $(r_1, \ldots, r_{k-1})$ provides the coarse-to-fine context for predicting r_k. At each autoregressive step k, all tokens in r_k are generated in parallel, conditioned on the previous scales and their positional embeddings.

3.2 OBS-Guided Structured Pruning for VAR

Network pruning reduces model size and inference cost by removing redundant or less important parameters. We focus on post-training structured pruning,



(a) Pyramid-Aware Hessian Accumulation (b) OBS-guided structured pruning (c) Scale-Aware Distillation

Fig. 3: The overall pipeline of EVAR. (a) Pyramid-Aware Hessian Accumulation builds a scale-aware Hessian from multi-scale calibration tokens. **(b)** OBS-guided structured pruning removes coupled attention heads and FFN channels with closed-form compensation. **(c)** PSAD balances multi-scale distillation with scale-aware weighting and a coarse-to-fine curriculum.

where entire channels or heads are removed so that the resulting model remains compatible with hardware acceleration. The central challenge is to reduce parameters while controlling the degradation in generation quality, especially for next-scale VAR models with long token sequences.

Layer-wise OBS formulation. Following Optimal Brain Surgeon (OBS), we decompose the global pruning problem into layer-wise subproblems and minimize the discrepancy between the original and pruned layer outputs in a local ℓ_2 sense. For the l-th layer with weight matrix W_l and input activation X_l, we consider the objective:

$$\min_{\hat{W}_l} \quad \left\| W_l X_l - \hat{W}_l X_l \right\|_2^2, \tag{2}$$

subject to a target pruning ratio on W_l. A second-order Taylor expansion around W_l yields a quadratic approximation of the loss in terms of the weight perturbation ΔW_l, with layer-wise Hessian:

$$H_l = 2X_l^\top X_l. \tag{3}$$

For individual weights in W_l, OBS selects the weight with the smallest estimated pruning-induced error and applies closed-form compensation to the re-

maining weights:

$$w_p^\star = \arg\min_{w_p} \frac{w_p^2}{(H_l^{-1})_{p,p}}, \qquad \delta_p = -\frac{w_p^\star}{(H_l^{-1})_{p,p}} (H_l^{-1})_{:,p}, \tag{4}$$

where $(H_l^{-1})_{p,p}$ and $(H_l^{-1})_{:,p}$ denote the p-th diagonal entry and column of H_l^{-1}, respectively. After pruning $w_p^\star$ to zero, δ_p is added to the remaining weights.

Structured OBS for VAR blocks. For next-scale VAR, we extend the above OBS formulation to structured units in the Transformer blocks, using calibration activations collected across all scales. Let $W^{(S)} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ denote a weight matrix in such a block, and let $H^{(S)} \in \mathbb{R}^{d_{\text{in}} \times d_{\text{in}}}$ denote the corresponding input Hessian estimate. We adopt column pruning as the basic operation and group columns into removable structured units, including attention-head dimensions in the attention output projection and intermediate channels in the FFN down-projection. For efficiency, we approximate the unit-wise OBS error by summing the column-wise OBS scores within each unit:

$$\text{score}(u) = \sum_{p \in u} \frac{\left\| W_{:,p}^{(S)} \right\|_2^2}{\left((H^{(S)})^{-1} \right)_{p,p}}, \quad \Delta^{(S)} = -\frac{W_{:,p}^{(S)}}{(H^{(S)})_{p,p}^{-1}} \, (H^{(S)})_{p,:}^{-1}. \tag{5}$$

where $(H^{(S)})_{p,:}^{-1}$ is the p-th row of $(H^{(S)})^{-1}$ and $\Delta^{(S)}$ is a compensation matrix with the same shape as $W^{(S)}$.

Directly applying stepwise column pruning with OBS is still expensive for VAR due to the large intermediate dimensions and the interdependence between attention heads. We therefore adopt an iterative unit-wise pruning-and-compensation scheme: (i) define attention heads and FFN channels as structured units; (ii) estimate the OBS error for each unit using the Hessian statistics from the Pyramid-Aware OBS method (Sec. 3.3); (iii) prune the least important unit and apply the compensation update in Eq. (5); and (iv) repeat until the target sparsity is reached. For FFN layers, we further use a dynamic grouped strategy that prunes small groups of low-score channels at a time, starting with larger groups and gradually decreasing the group size as pruning progresses. Empirically, this structured OBS scheme provides strong compression for VAR while maintaining high image generation quality.

3.3 Pyramid-Aware Hessian Accumulation for Next-Scale VAR

OBS-guided pruning relies critically on a precise calibration set to construct the layer-wise Hessian estimates H_l. To better approximate the inference-time token distribution, we adopt a pre-encoded calibration strategy. Specifically, we encode real images using the VQ-VAE encoder and the residual pyramid used by VAR to obtain exact multi-scale calibration tokens, avoiding the bias introduced by model-generated tokens. However, even with such accurate multi-scale tokens,

the conventional OBS-based calibration paradigm remains structurally inadequate for next-scale VAR models, because it simply concatenates tokens from all scales when accumulating the Hessian.

This inadequacy stems from two core characteristics of the multi-scale generation paradigm. First, Hessian estimation bias driven by scale-wise token imbalance: coarse-scale tokens dictate the global structural quality of the image, yet the token sequence length grows dramatically at later scales. Under standard Hessian accumulation, the resulting Hessian estimate is dominated by abundant late-scale tokens, thereby underrepresenting the crucial early scales and leading to biased saliency estimates. Therefore, the flat token-assembly strategy commonly used in LLM pruning is not well suited to next-scale VAR models. Second, error cascading: due to the next-scale autoregressive nature, any pruning-induced errors at early scales can propagate through subsequent scales and be amplified in the final image.

To address this issue, we propose Pyramid-Aware OBS, which constructs a scale-aware Hessian estimate. A naive workaround to protect early scales would be to split the multi-scale tokens and compute Hessians independently for each scale. However, this leads to severe rank deficiency in the resulting covariance estimate: for small scales such as 1×1 or 2×2, the number of tokens is far smaller than the hidden dimension D. Using only these tokens yields a singular covariance matrix, precluding the matrix inversion required by OBS. In contrast, our pyramid-aware Hessian is constructed as a weighted Gram matrix over calibration activations. When all scale weights are strictly positive, this construction is positive semi-definite by construction, but it is not necessarily invertible. If the weighted concatenation of calibration activations across all scales has full column rank, the undamped Gram matrix becomes positive definite. In practice, we further add a damping term λI, which guarantees positive definiteness and stabilizes the OBS compensation. This also explains why independent scale-wise Hessian estimation is problematic: the activations from individual early scales alone cannot span the full feature space. To collect sufficiently diverse activations without excessive memory cost, we use a streaming calibration pipeline. By processing fixed-size image batches and discarding intermediate activations after accumulation, we can aggregate activation statistics across scales and calibration batches without excessive memory overhead.

To rebalance the scale-wise contribution while maintaining a stable inverse for OBS compensation, we apply scale-aware decay weighting to the normalized Hessian accumulation.

$$H_{\mathrm{pa}} = \sum_{i=1}^{K} \frac{w_i}{n_i} \left(X^{(i)}\right)^{\top} X^{(i)} + \lambda I. \tag{6}$$

where $X^{(i)}$ denotes the input activations for the n_i tokens at scale i. The factor $1/n_i$ counteracts the token-count disparity across scales, while the scale-decay weight w_i assigns larger relative importance to early scales. Consequently, the OBS saliency and closed-form compensation computed with H_{pa}^{-1} place greater emphasis on weight directions that affect the highly weighted early scales.

3.4 Progressive Scale-Aware Distillation

Although vanilla knowledge distillation can recover part of the post-pruning performance, it does not explicitly account for the generative hierarchy of next-scale VAR. Since the token sequence length grows quadratically at later scales, unweighted distillation naturally induces a scale-wise gradient imbalance. As a result, the objective is biased toward abundant high-resolution tokens, providing insufficient supervision for coarse-scale semantics and global layout. Because pruning reduces the representational capacity of the model, this bias further hinders the compressed network from preserving global structure.

To address this imbalance, we introduce Progressive Scale-Aware Distillation (PSAD). PSAD combines a progressive coarse-to-fine curriculum with scale-aware distillation weighting to explicitly rebalance the multi-scale gradient distribution, so that coarse-scale representations that determine global structure receive sufficient supervision.

Setup. Let K be the total number of scales, and L the flattened sequence length. Scales are indexed by $i \in \{1, \dots, K\}$. We define the cumulative token boundaries as $c_0 = 0$ and $c_K = L$, where scale i contains tokens from $c_{i-1} + 1$ to c_i. The number of tokens at scale i is therefore $n_i = c_i - c_{i-1}$. Given a distillation temperature $\tau > 0$ and batch size B, let $p_T^{(m,\ell)}$ and $p_S^{(m,\ell)}$ denote the temperature-softened output distributions of the dense teacher and the pruned student at token ℓ of sample m, respectively. The scale-level KL divergence is defined as:

$$\mathcal{L}_{\mathrm{KL}}^{(i)} = \frac{1}{B} \sum_{m=1}^{B} \sum_{\ell=c_{i-1}+1}^{c_i} \mathrm{KL}\Big(p_T^{(m,\ell)} \,\|\, p_S^{(m,\ell)}\Big). \tag{7}$$

We scale this loss by τ^2 to compensate for the temperature-induced change in gradient magnitude.

1) Progressive Coarse-to-Fine Curriculum. To prevent high-resolution tokens from dominating the early recovery stage, we progressively activate scales during training. We define a continuous gate $\tilde{\gamma}_i(t)$ to smoothly activate scale i at optimization step t:

$$\tilde{\gamma}_i(t) = \min\left\{1,\ \max\left\{0,\ \frac{t - t_i}{\Delta_i}\right\}\right\}, \tag{8}$$

where t_i denotes the step at which scale i starts to be activated, and $\Delta_i > 0$ controls the ramp-up duration. Compared with discrete scale unlocking, this soft gate avoids abrupt changes in the optimization objective and improves the training stability of capacity-limited pruned models. Accordingly, the task loss $\mathcal{L}_{\mathrm{task}}(t)$ is computed with the same scale gates, so that inactive scales are excluded and partially activated scales contribute proportionally.

2) Scale-Aware Distillation Weighting and Gradient Balancing. To mitigate the gradient dominance of high-resolution scales, we define a scale-aware distillation weight w_i that is inversely proportional to the number of tokens at each scale:

$$w_i = \frac{C}{n_i}. \tag{9}$$

where C is a normalization constant used to keep the overall loss scale stable. The PSAD loss is then formulated as:

$$\mathcal{L}_{\mathrm{PSAD}}(t) = \tau^2 \sum_{i=1}^{K} \tilde{\gamma}_i(t)\, w_i\, \mathcal{L}_{\mathrm{KL}}^{(i)}. \tag{10}$$

Let $\bar{g}_i$ denote the average per-token gradient norm at scale i. Since $\mathcal{L}_{\mathrm{KL}}^{(i)}$ sums over n_i tokens, the effective gradient contribution of scale i approximately scales as $g_i(t) \propto \tilde{\gamma}_i(t)\, w_i\, n_i\, \bar{g}_i$. Substituting $w_i = C/n_i$ cancels the token-count factor, reducing $w_i n_i$ to the constant C. This counteracts the token-count advantage of fine scales and yields more balanced gradient contributions across scales. The final training objective is defined as:

$$\mathcal{L}_{\mathrm{total}}(t) = \mathcal{L}_{\mathrm{task}}(t) + \mathcal{L}_{\mathrm{PSAD}}(t). \tag{11}$$

4 Experimental Results

4.1 Experimental Setups

We evaluate EVAR on the class-conditional ImageNet 256×256 generation benchmark using VAR as the base backbone, and compare it with state-of-the-art autoregressive and VAR-based image generation models. For model calibration, we use class-balanced sampling from the ImageNet training set, selecting one real image per class for streaming Hessian accumulation. We empirically find that using substantially larger calibration sets brings only marginal gains in pruning performance. Following pruning, all compressed models undergo post-training adaptation for 40 epochs. We use the AdamW optimizer and follow the original VAR training hyperparameters, including the learning-rate schedule and weight-decay annealing. Fine-tuning is performed on a single node with 8 NVIDIA RTX 5090 GPUs, taking approximately one hour per epoch for the pruned VAR-d16.

For edge deployment, the pruned models are converted to Core ML and deployed on an iPad Pro (M4), where we measure the on-device latency of single-image generation.

4.2 Main Results

We evaluate EVAR from two perspectives: generation quality after compression and pruning effectiveness against representative structured pruning baselines. Table 1 reports parameter counts, pruning rates, and generation quality before

Table 1: Generative performance on class-conditional ImageNet-256. "Steps" denotes the number of model inference steps required to generate one image.

Type	Model	Parameters	Pruning Rate	Steps	FID ↓	IS↑	Precision↑	Recall↑
VAR	VAR-d16 [44]	310M	–	10	3.55	274.4	0.84	0.51
	VAR-d20 [44]	600M	–	10	2.95	302.6	0.83	0.56
	VAR-d24 [44]	1.0B	–	10	2.33	312.9	0.82	0.59
AR	VQGAN-re [9]	1.4B	–	256	5.20	280.3	–	–
	RQ-Trans.-re [21]	3.8B	–	64	3.80	323.7	–	–
	LlamaGen-L [42]	343M	–	576	3.07	256.1	0.83	0.52
	LlamaGen-XL [42]	775M	–	576	2.62	244.1	0.80	0.57
	LlamaGen-XXL [42]	1.4B	–	576	2.62	244.1	0.80	0.57
	CoDe(N=8) [8]	2.3B	–	10	2.26	300.4	0.81	0.58
	FastVAR [15](d24)	1B	–	10	2.64	284.4	0.80	0.58
	PAR-XXL [52]	1.4B	–	147	2.35	263.2	0.80	0.62
	AiM-L [23]	350M	–	256	2.83	244.6	0.82	0.55
	AiM-XL [23]	763M	–	256	2.56	257.2	0.82	0.57
VAR-d16	EVAR (ours)	270M	20%	10	3.67	267.5	0.81	0.51
	EVAR (ours)	230M	40%	10	3.91	259.1	0.81	0.51
VAR-d24	EVAR (ours)	875M	20%	10	2.47	310.2	0.82	0.57
	EVAR (ours)	748M	40%	10	2.52	304.4	0.82	0.57

and after pruning. Within the VAR family, we evaluate EVAR on both VAR-d16 and VAR-d24 to examine its scalability across model sizes. For VAR-d16, EVAR reduces the parameter count from 310M to 270M at 20% sparsity with a FID of 3.67, close to the dense baseline's 3.55. At 40% sparsity, it further compresses the model to 230M parameters while maintaining a competitive FID of 3.91. For VAR-d24, EVAR achieves similar robustness: the 20% and 40% pruned models reach 875M and 748M parameters with FIDs of 2.47 and 2.52, respectively. These results indicate that EVAR remains effective across different VAR backbone scales, achieving substantial compression with only modest degradation in generation quality.

Because existing structured pruning methods do not report results on next-scale VAR, we reimplement representative pruning baselines on VAR-d16 under a unified evaluation protocol. All methods are applied to the same backbone and evaluated under matched sparsity levels, fine-tuning settings, and ImageNet-256 evaluation metrics.

As summarized in Tab. 2, EVAR (ours) consistently outperforms a range of structured pruning baselines across sparsity levels and fine-tuning regimes. Under the most challenging setting of 40% sparsity without any fine-tuning, all methods suffer large quality drops, but EVAR still achieves the best FID/IS (58.21 / 21.12), whereas other methods degrade much more severely. At 20% sparsity in the training-free regime, EVAR attains a substantially lower FID of 7.85 with strong precision/recall, while all baselines remain in a much worse range (FID 27.30–152.29), indicating that OBS-guided pruning with pyramid-

aware Hessian accumulation is particularly effective for moderate compression. Although SlimGPT adopts OBS-style structured pruning, it estimates the Hessian using an LLM-oriented calibration strategy and does not account for VAR's multi-scale token hierarchy. Under the same VAR pruning setting, its performance is worse than EVAR, e.g., FID 56.87 versus 7.85 at 20% sparsity. This suggests that OBS-style compensation alone is insufficient for next-scale VAR, and that the Hessian estimate must account for the multi-scale token hierarchy.

Furthermore, to evaluate post-pruning recovery, we fine-tune all models for only one epoch under the challenging 40% sparsity setting. While the baselines still suffer from noticeable quality degradation, with FID scores between 8.35 and 12.69, EVAR reaches the best FID of 4.86 and the highest IS of 55.61. These results indicate that Pyramid-Aware OBS better preserves coarse-scale representations during pruning, providing a stronger post-pruning initialization and enabling more effective recovery with limited fine-tuning.

Table 2: Comparison of pruning methods under different sparsity and fine-tuning regimes.

	40% sparsity, training-free				20% sparsity, training-free				40% sparsity, 1-epoch			
Method	FID↓	IS ↑	Precision↑	Recall ↑	FID↓	IS ↑	Precision↑	Recall↑	FID↓	IS ↑	Precision↑	Recall↑
SlimGPT [30]	138.91	6.39	0.09	0.07	56.87	19.35	0.36	0.57	10.62	44.91	0.77	0.46
Magnitude [17]	180.82	3.90	0.17	0.01	152.29	5.77	0.10	0.01	12.69	25.84	0.65	0.38
LLM-pruner [33]	153.82	5.62	0.09	0.02	35.21	28.95	0.41	0.51	10.82	41.35	0.76	0.45
OBA [41]	142.82	6.35	0.12	0.04	30.54	29.48	0.45	0.57	9.85	44.35	0.78	0.46
Taylor [34]	145.66	6.78	0.12	0.05	27.30	31.48	0.48	0.59	8.35	46.87	0.77	0.46
EVAR (ours)	58.21	21.12	0.35	0.53	7.85	49.21	0.74	0.51	4.86	55.61	0.82	0.47

Table 3: Ablation of calibration and Hessian accumulation strategies for pruning VAR-d16 at 40% sparsity. First/Fifth/Last scale denote selecting tokens from a specific scale as the calibration set, while Direct refers to directly concatenating tokens across scales.

Method	**FID↓**	**Params (M)**
Dense	3.55	310
First scale	124.21	230
Fifth scale	102.54	230
Last scale	86.51	230
Direct	69.21	230
Ours	**58.21**	230

Table 4: Ablation of distillation components and scale-wise attention strategies in EVAR. All pruned variants maintain the same 230M parameter budget.

Method	**FID↓**	**Params (M)**
Dense	3.55	310
Pruned (training-free)	58.21	230
+ Fine-tuning (FT)	4.25	230
+ Vanilla KD	4.26	230
+ Exponential Increase	5.82	230
+ Linear Increase	5.45	230
+ Exponential Decay	4.36	230
+ PSAD (ours)	**3.91**	230

4.3 Ablation Study

Table 3 reports an ablation study on calibration and Hessian accumulation strategies for pruning VAR-d16 at 40% sparsity. Using tokens from a single scale leads to severe degradation in generation quality, with FID ranging from 86.51 to 124.21, indicating that a single-scale calibration source cannot adequately represent VAR's full multi-scale generation hierarchy. Directly concatenating tokens from all scales provides broader calibration coverage and improves FID to 69.21 over the single-scale variants, but it still suffers from scale-wise token imbalance. However, our proposed calibration strategy achieves the best performance (FID 58.21). This suggests that the proposed pyramid-aware accumulation better balances multi-scale calibration statistics and yields a more reliable Hessian estimate under the same 230M parameter budget.

We ablate the distillation components in EVAR to verify our scale-aware strategy. As shown in Tab. 4, training-free pruning causes substantial quality degradation (FID 58.21). Plain fine-tuning and vanilla knowledge distillation (KD) recover most of the lost quality (FID $\approx$ 4.25), but their uniform token-level objectives still bias optimization toward fine-scale tokens due to the much larger sequence length at later scales. Among different weighting strategies, those assigning higher weights to later scales, namely Linear Increase and Exponential Increase, lead to worse performance, with FIDs of 5.45 and 5.82, respectively. This is consistent with our hypothesis: amplifying the gradient dominance by high-resolution tokens causes the model to neglect the global semantics dictated by coarse scales. Conversely, a naive Exponential Decay over-corrects, resulting in a worse FID of 4.36 (worse than Vanilla KD). Exponential decay suppresses fine-scale gradients too aggressively, leading to gradient starvation for local texture refinement. In contrast, our Progressive Scale-Aware Distillation (PSAD) employs an inverse-proportional decay ($w_i \propto 1/n_i$) coupled with a progressive coarse-to-fine curriculum. This counteracts token-count dominance while preserving sufficient supervision for fine scales, rebalancing the multi-scale gradient distribution. PSAD yields the best result (FID 3.91), nearly closing the gap to the dense baseline under the same 277 ms / 230M edge deployment configuration.

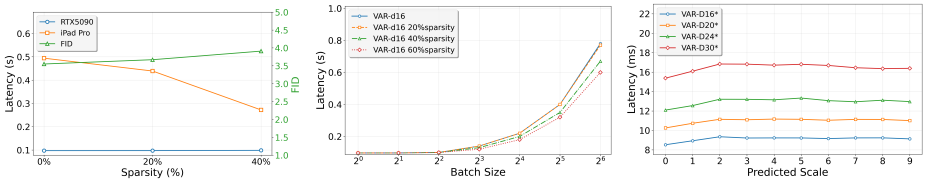
4.4 Edge Deployment

Core ML conversion and runtime. For on-device evaluation, we convert the pruned VAR-d16 models from PyTorch to Core ML `.mlmodel` format using `coremltools`, and deploy them in a Swift-based single-image generation app on an iPad Pro (M4). The Core ML runtime assigns operators to available compute units, including the CPU, GPU, and Neural Engine (NPU), with fallbacks when certain operators are unsupported on a selected unit.

Compute-unit behavior. Next-scale VAR involves several positional-embedding and sampling-related operations that are not efficiently assigned to the NPU, so enabling the NPU can introduce frequent fallbacks and cross-unit transitions among the CPU, GPU, and NPU. Table 5 reports an operator-level breakdown

Table 5: Operator-level unit assignment in Core ML for EVAR on iPad Pro (M4). Frequent switches across CPU/GPU/NPU due to NPU-unsupported ops inflate latency; pinning to CPU+GPU is fastest for single-image inference.

Device	Model	Compute units	Latency (ms)	#CPU ops	#GPU ops	#NPU ops
iPad Pro (M4)	EVAR	CPU+NPU	1090 ± 2	274	0	8554
iPad Pro (M4)	EVAR	All	494 ± 2	129	3103	5596
iPad Pro (M4)	EVAR	CPU+GPU	277 ± 2	10	8818	0



(a) Latency–sparsity trade-off on RTX 5090 and iPad Pro (M4).

(b) Latency vs. batch size on RTX 5090 for different sparsity levels.

(c) Per-scale single-image latency on RTX 5090 across VAR depths.

Fig. 4: Latency behavior of EVAR under different sparsity and deployment settings.

for EVAR on an iPad Pro (M4) under three Core ML settings: CPU+GPU, CPU+NPU, and All. Although CPU+NPU assigns most operators to the NPU, it still leaves 274 operators on the CPU, causing frequent CPU–NPU transitions and yielding the highest latency (1090 ± 2 ms). The All setting mixes all three units (129 CPU ops, 3103 GPU ops, 5596 NPU ops) and is also slower (494 ± 2 ms) due to frequent cross-unit transitions. In contrast, pinning execution to CPU+GPU avoids NPU-related fallbacks, yields a simpler two-unit schedule (10 CPU ops and 8818 GPU ops), and achieves the lowest latency (277 ± 2 ms) for single-image inference.

Latency analysis. Fig. 4(a) shows that increasing sparsity barely changes single-image latency on the RTX 5090, but substantially reduces latency on the iPad Pro from 494 ms to 277 ms with only a mild FID increase, showing that EVAR is highly effective in the edge compute-bound regime. More importantly, when deployed on a mobile phone (iPhone 16 Pro Max), the dense baseline fails to execute due to strict memory constraints, whereas our 40% pruned model runs smoothly. This indicates that compression is important not only for acceleration, but also for improving the deployability of next-scale VAR models on memory-constrained mobile devices. More comprehensive on-device results can be found in the supplementary material. Fig. 4(b) shows that on the RTX 5090, dense and pruned VAR-d16 models have almost identical latency at batch size 1, while their latency differences become more visible at larger batch sizes, suggesting that single-image inference on the GPU is largely memory-bound. Fig. 4(c) shows the per-scale single-image latency on the RTX 5090. The latency curves are nearly flat across predicted scales, even though later scales contain more tokens,

indicating that server-side batch-size-one inference is largely memory-bound or overhead-dominated. This contrasts with mobile deployment, where limited parallel compute makes inference more compute-bound and allows EVAR's width reduction to translate into clear latency gains.

5 Conclusion

This paper introduces EVAR, a principled structured-pruning framework tailored to next-scale visual autoregressive (VAR) models. Motivated by the mismatch between standard pruning paradigms and next-scale VAR models, where scale-wise token imbalance and error cascading can degrade pruning reliability, EVAR introduces Pyramid-Aware OBS. By using scale-normalized Hessian accumulation with scale-aware decay weighting, Pyramid-Aware OBS better preserves early-scale representations while maintaining the positive definiteness required for closed-form OBS compensation. To optimize post-pruning recovery, we introduce Progressive Scale-Aware Distillation (PSAD). By coupling inverse-proportional scale weighting with a coarse-to-fine curriculum, PSAD counteracts the token-count dominance of fine scales and balances gradient contributions across the multi-scale generation hierarchy. Experiments on ImageNet 256×256 show that EVAR reduces model parameters while maintaining competitive generation fidelity. Specifically, our iOS deployment of a pruned VAR-d16 model attains a **1.8× on-device speedup** with marginal FID degradation. Overall, EVAR shows that next-scale VAR models can be systematically compressed for efficient edge execution, making on-device visual synthesis more practical.

Acknowledgement

This paper is supported by Young Scientists Fund of the National Natural Science Foundation of China (NSFC) (No. 62506305), and in part by Zhejiang Leading Innovative and Entrepreneur Team Introduction Program (No. 2024R01007), and in part by Key Research and Development Program of Zhejiang Province (No. 2025C01026), and in part by Scientific Research Project of Westlake University (No. WU2025WF003). It is also supported in part by the National Natural Science Foundation of China under Grant 62303405, and in part by Ningbo Natural Science Foundation Project under Grant 2025Z105.

References

1. Ashkboos, S., Croci, M.L., Nascimento, M.G.d., Hoefler, T., Hensman, J.: Slicegpt: Compress large language models by deleting rows and columns. In: ICLR (2024)
2. Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D.: Language models are few-shot learners. arXiv preprint arXiv:2005.14165 (2020)

3. Chang, H., Zhang, H., Jiang, L., Liu, C., Freeman, W.T.: Maskgit: Masked generative image transformer. In: CVPR (2022)
4. Chen, J., Hu, D., Huang, X., Coskun, H., Sahni, A., Gupta, A., Goyal, A., Lahiri, D., Singh, R., Idelbayev, Y., et al.: Snapgen: Taming high-resolution text-to-image models for mobile devices with efficient architectures and training. In: CVPR (2025)
5. Chen, T., Liang, L., Ding, T., Zhu, Z., Zharkov, I.: Otov2: Automatic, generic, user-friendly. In: ICLR (2023)
6. Chen, Y., Wang, Z.: An effective information theoretic framework for channel pruning. arXiv preprint arXiv:2408.16772 (2024)
7. Chen, Y., Lan, Y., Zhou, S., Wang, T., Pan, X.: Sar3d: Autoregressive 3d object generation and understanding via multi-scale 3d vqvae. In: CVPR (2025)
8. Chen, Z., Ma, X., Fang, G., Wang, X.: Collaborative decoding makes visual autoregressive modeling efficient. In: CVPR (2025)
9. Esser, P., Rombach, R., Ommer, B.: Taming transformers for high-resolution image synthesis. In: CVPR (2021)
10. Fang, G., Ma, X., Song, M., Mi, M.B., Wang, X.: Depgraph: Towards any structural pruning. In: CVPR (2023)
11. Feng, S., Tao, K., Wang, H.: Is oracle pruning the true oracle? TMLR (2026)
12. Frantar, E., Alistarh, D.: Optimal brain compression: A framework for accurate post-training quantization and pruning. In: NeurIPS (2022)
13. Frantar, E., Alistarh, D.: Sparsegpt: Massive language models can be accurately pruned in one-shot. In: ICML (2023)
14. Gu, J., Wang, Y., Zhang, Y., Zhang, Q., Zhang, D., Jaitly, N., Susskind, J.M., Zhai, S.: Denoising autoregressive transformers for scalable text-to-image generation. In: ICLR (2025)
15. 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: CVPR. pp. 19011–19021 (2025)
16. 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: CVPR (2025)
17. Han, S., Pool, J., Tran, J., Dally, W.J.: Learning both weights and connections for efficient neural networks. In: NeurIPS (2015)
18. Hassibi, B., Stork, D., Wolff, G.: Optimal brain surgeon and general network pruning. In: ICNN (1993)
19. He, Y., Zhang, X., Sun, J.: Channel pruning for accelerating very deep neural networks. In: ICCV (2017)
20. LeCun, Y., Denker, J.S., Solla, S.A.: Optimal brain damage. In: NeurIPS (1990)
21. Lee, D., Kim, C., Kim, S., Cho, M., Han, W.S.: Autoregressive image generation using residual quantization. arXiv preprint arXiv:2203.01941 (2022)
22. Li, H., Kadav, A., Durdanovic, I., Samet, H., Graf, H.P.: Pruning filters for efficient convnets. In: ICLR (2017)
23. Li, H., Yang, J., Wang, K., Qiu, X., Chou, Y., Li, X., Li, G.: Scalable autoregressive image generation with mamba. arXiv preprint arXiv:2408.12245 (2024)
24. Li, T., Tian, Y., Li, H., Deng, M., He, K.: Autoregressive image generation without vector quantization. arXiv preprint arXiv:2406.11838 (2024)
25. Li, X., Qiu, K., Chen, H., Kuen, J., Gu, J., Raj, B., Lin, Z.: Imagefolder: Autoregressive image generation with folded tokens. In: ICLR (2025)

26. Li, X., Qiu, K., Chen, H., Kuen, J., Lin, Z., Singh, R., Raj, B.: Controlvar: Exploring controllable visual autoregressive modeling. arXiv preprint arXiv:2406.09750 (2024)
27. Li, Y., Wang, H., Jin, Q., Hu, J., Chemerys, P., Fu, Y., Wang, Y., Tulyakov, S., Ren, J.: Snapfusion: Text-to-image diffusion model on mobile devices within two seconds. In: NeurIPS (2023)
28. Li, Y., Lv, C., Wang, H.: Freqexit: Enabling early-exit inference for visual autoregressive models via frequency-aware guidance. In: NeurIPS (2025)
29. Liao, B., Li, Y., Jian, S., Wang, H.: Parallel jacobi decoding for fast autoregressive image generation. In: CVPR. pp. 9008–9018 (2026)
30. Ling, G., Wang, Z., Yan, Y., Liu, Q.: Slimgpt: Layer-wise structured pruning for large language models. In: NeurIPS (2024)
31. Liu, D., Zhao, S., Zhuo, L., Lin, W., Xin, Y., Li, X., Qin, Q., Qiao, Y., Li, H., Gao, P.: Lumina-mgpt: Illuminate flexible photorealistic text-to-image generation with multimodal generative pretraining. arXiv preprint arXiv:2408.02657 (2024)
32. Liu, E., Ning, X., Wang, Y., Lin, Z.: Distilled decoding 1: One-step sampling of image auto-regressive models with flow matching. In: ICLR (2025)
33. Ma, X., Fang, G., Wang, X.: Llm-pruner: On the structural pruning of large language models. In: NeurIPS (2023)
34. Molchanov, P., Mallya, A., Tyree, S., Frosio, I., Kautz, J.: Importance estimation for neural network pruning. In: CVPR (2019)
35. Radford, A., Narasimhan, K.: Improving language understanding by generative pre-training. Semantic Scholar (2018)
36. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)
37. 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)
38. Singh, S.P., Alistarh, D.: Woodfisher: Efficient second-order approximation for neural network compression. In: NeurIPS (2020)
39. Su, M., Wang, H.: Rose: Reordered sparsegpt for more accurate one-shot large language models pruning. In: CPAL (2026)
40. Sun, M., Liu, Z., Bair, A., Kolter, J.Z.: A simple and effective pruning approach for large language models. In: ICLR (2024)
41. Sun, M., Fang, Z., Wang, J., Jiang, J., Kong, D., Hu, C., Fang, Y., Xu, R.: Optimal brain apoptosis. In: ICLR (2025)
42. 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)
43. 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. In: ICLR (2025)
44. Tian, K., Jiang, Y., Yuan, Z., Peng, B., Wang, L.: Visual autoregressive modeling: Scalable image generation via next-scale prediction. In: NeurIPS (2024)
45. Tu, X., He, Z., Huang, Y., Zhang, Z.H., Yang, M., Zhao, J.: An overview of large ai models and their applications. Visual Intelligence **2**(1), 34 (2024)
46. Van Den Oord, A., Kalchbrenner, N., Kavukcuoglu, K.: Pixel recurrent neural networks. In: ICML (2016)
47. Van Den Oord, A., Kalchbrenner, N., Vinyals, O., Espeholt, L., Graves, A., Kavukcuoglu, K.: Conditional image generation with pixelcnn decoders. In: NeurIPS (2016)

48. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017)
49. Wang, H., Qin, C., Zhang, Y., Fu, Y.: Neural pruning via growing regularization. In: ICLR (2021)
50. Wang, H., Zhang, Y., Qin, C., Van Gool, L., Fu, Y.: Global aligned structured sparsity learning for efficient image super-resolution. IEEE transactions on pattern analysis and machine intelligence **45**(9), 10974–10989 (2023)
51. Wang, J., Liu, J., Tang, D., Wang, W., Li, W., Chen, D., Chen, J., Wu, J.: Scalable autoregressive monocular depth estimation. In: CVPR (2025)
52. Wang, Y., Ren, S., Lin, Z., Han, Y., Guo, H., Yang, Z., Zou, D., Feng, J., Liu, X.: Parallelized autoregressive visual generation. arXiv preprint arXiv:2412.15119 (2024)
53. Xiaoxiao, M., Mohan, Z., Tao, L., Yalong, B., Tiejun, Z., Biye, L., Huaian, C., Yi, J.: Star: Scale-wise text-conditioned autoregressive image generation. arXiv preprint arXiv:2406.10797 (2024)
54. Xie, R., Zhao, T., Yuan, Z., Wan, R., Gao, W., Zhu, Z., Ning, X., Wang, Y.: Litevar: Compressing visual autoregressive modelling with efficient attention and quantization. arXiv preprint arXiv:2411.17178 (2024)
55. Zhang, Q., Dai, X., Yang, N., An, X., Feng, Z., Ren, X.: Var-clip: Text-to-image generator with visual auto-regressive modeling. arXiv preprint arXiv:2408.01181 (2024)
56. Zhou, C., Yu, L., Babu, A., Tirumala, K., Yasunaga, M., Shamis, L., Kahn, J., Ma, X., Zettlemoyer, L., Levy, O.: Transfusion: Predict the next token and diffuse images with one multi-modal model. arXiv preprint arXiv:2408.11039 (2024)
57. Zhu, J., Wang, H., Su, M., Wang, Z., Wang, H.: Obs-diff: Accurate pruning for diffusion models in one-shot. In: ICLR (2026)