DYNAMIC TOKENIZATION FOR FLEXIBLE IMAGE REPRESENTATION AND GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing image tokenizers typically use a fixed number of tokens at fixed spatial locations, overlooking the substantial variation in visual complexity both across images and within individual images. To address this limitation, we introduce **DynamicTok** (Dynamic Image Tokenizer), which adaptively allocates tokens in both quantity and spatial location according to content complexity. DynamicTok first predicts a heatmap that estimates regional importance, assigning more tokens to visually informative regions and fewer tokens to redundant ones. To process these spatially non-uniform tokens effectively, we propose a hybrid 1D–2D tokenizer in which latent tokens capture global context while receiving localized cues from image tokens. DynamicTok can be readily integrated with existing generative models. Moreover, we introduce a two-stage generation strategy that predicts a heatmap before image synthesis, enabling flexible generation under different token budgets. On ImageNet-256 reconstruction, DynamicTok achieves rFID scores of 0.57 and 0.39 with 64 and 128 tokens, respectively, outperforming prior methods under the same token budgets. For ImageNet-256 generation, DynamicTok attains a gFID of 1.40 with 64 tokens, establishing a new state of the art among methods using fewer than 256 tokens. Moreover, both our tokenizer and generator can dynamically allocate tokens in terms of number and spatial locations as needed.

1 INTRODUCTION

Latent image generative models Rombach et al. (2022); Peebles & Xie (2023); Esser et al. (2024; 2021); Sun et al. (2024); Tian et al. (2024) employ image tokenizers to compress images into latent representations, enabling generative training in a compact latent space. This design significantly reduces computational cost and enable high-quality image synthesis. Most existing approaches, following Stable Diffusion Rombach et al. (2022), utilize a continuous or discrete 2D autoencoder to transform images into grid-structured 2D latent tokens. However, this fixed grid representation neglects the fact that both images and local regions exhibit varying levels of inherent visual complexity. Moreover, different scenes may emphasize distinct regions and have varying computational resources, thereby requiring both spatial and quantitative adaptivity for more flexible tokenization and generation.

Recent studies such as FlexTok Bachmann et al. (2025) and One-D-Piece Miwa et al. (2025) introduce 1D image tokenizers with Matryoshka learning Kusupati et al. (2022) to support dynamic token counts, while GaussianTok Dong et al. (2025) and GPSTok Zhang et al. (2025) incorporate learnable 2D Gaussian attributes to enhance spatial adaptability. Nevertheless, none of these methods simultaneously achieve both quantity and spatial adaptivity.

In this work, we propose to assign a variable number of tokens to different images and local regions. As illustrated in Fig. 1 (left), images are divided into non-uniform regions, each corresponding to a token. Simpler images or regions are allocated fewer tokens to reduce redundancy, whereas more complex ones receive more tokens to ensure sufficient representational capacity.

Estimating image and regional complexity is non-trivial. Prior work uses gradient maps Zhang et al. (2025) or learned predictors Huang et al. (2023); Jia et al. (2025) for token allocation. We find that simple gradient maps are sufficient to estimate structural complexity and achieve strong empirical performance. During training, we use the mean gradient magnitude to determine each image's token budget, then downsample and normalize the gradient map into a low-resolution heatmap (e.g., $16 \times 16 \times 1$) for partitioning regions with approximately uniform cumulative values. During

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

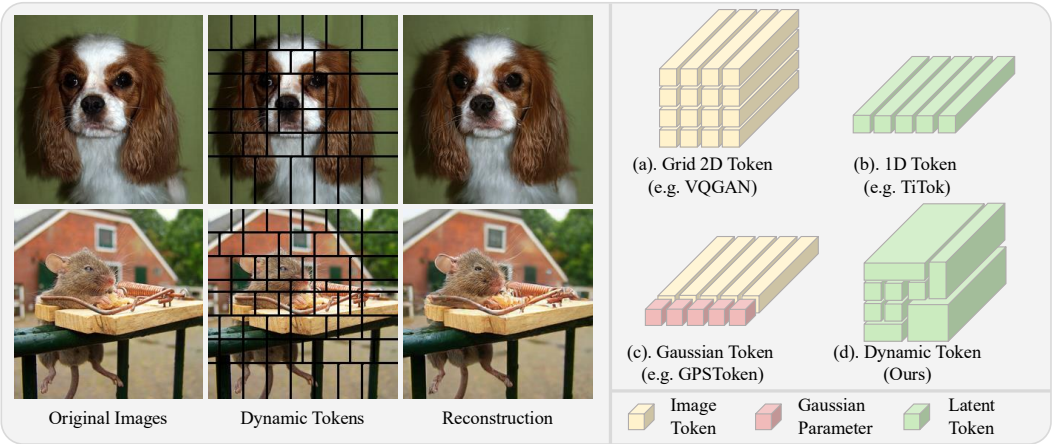


Figure 1: Left: DynamicTok assigns different numbers of tokens to different images and local regions.Right: Comparison of various tokenizer paradigms. (a) Traditional 2D tokenizers encode images into grid-structured tokens. (b) 1D tokenizers convert images into unstructured latent sequences. (c) Gaussian-parameterized tokenizers endow each token with learnable Gaussian attributes. (d) Our DynamicTok divides images into non-uniform patches and supports dynamic token counts.

inference, the same strategy can be used, or token counts and partitions can be adjusted according to computational constraints or task requirements.

To effectively process tokens with dynamic quantities and non-uniform spatial distributions, we propose a hybrid tokenizer between 1D and 2D designs. Conventional 1D tokenizers Yu et al. (2024a) typically adopt a Q-Former Li et al. (2023) structure, where query tokens attend to the global visual context, while 2D tokenizers Rombach et al. (2022); Esser et al. (2021) commonly rely on convolutional networks to extract uniform local features. In contrast, our approach builds upon the 1D tokenizer and introduces a fusion stage that injects local image features into the corresponding query tokens. Specifically, in each encoder layer, image patches within the same region are pooled into a single token, and these pooled tokens are fused with the query tokens to enhance spatial awareness. The decoder mirrors this process: query tokens are expanded back into a grid structure according to the partition information, and the expanded tokens are fused with masked image patches.

As for generation, DynamicTok can be seamlessly integrated with existing generators by using grid partitioning when extracting latent tokens. To support adaptive-length image synthesis, we adopt a two-stage generation pipeline that predicts the heatmap in the first stage and uses it to determine the token count and partition for image synthesis.

DynamicTok supports flexible tokenization and generation with dynamic token counts and spatial locations. As few as 32 tokens are sufficient for reconstructing and generating simple images, while more tokens can be allocated to detail-rich images and regions. DynamicTok also delivers strong performance in both reconstruction and generation. On the ImageNet 256×256 reconstruction task, DynamicTok achieves an rFID of 0.57 with 64 tokens and 0.39 with 128 tokens, surpassing previous methods under the same token budgets. For image generation, it attains a gFID of 1.40 using only 64 tokens, approaching the performance of previous state-of-the-art models that require 256 tokens.

2 Related Work

Image Tokenization. Early visual tokenization approaches are rooted in autoencoding frameworks, where image reconstruction serves as a self-supervised objective. Variational Autoencoders (VAEs) Kingma & Welling (2013) introduce a probabilistic latent space, while VQ-VAEs Van Den Oord et al. (2017); Razavi et al. (2019) further discretize the latent representation through vector quantization to form a visual codebook. VQGAN Esser et al. (2021) improves both reconstruction quality and semantic coherence by integrating adversarial learning. These methods typically adopt convolutional neural networks as the encoder–decoder backbone to capture local spatial correlations, while some works explore Transformer-based tokenizers Chang et al. (2022); Qu et al. (2025); Du et al. (2024); Yan et al. (2025); Zheng et al. (2026).Regardless of architectural differences, conven-

tional tokenizers assign each token to a fixed spatial region, leading to a 2D grid-like structure that closely reflects the original image layout.

Recent studies Yu et al. (2024a); Duggal et al. (2024); Chen et al. (2025b) have proposed 1D tokenizers that represent the entire image as a compact 1D token sequence to alleviate redundancy in the 2D tokenizers. TiTok Yu et al. (2024a) design an additional set of learnable latent tokens that interact with image features through cross-attention, enabling flexible compression and significantly reducing token redundancy. Following this direction, SoftVQ Chen et al. (2025b) increases the representation capacity of the latent space by adaptive aggregation of multiple codewords.

In parallel, some methods investigate the dynamics of tokenization, focusing on adaptivity in token number or spatial position. To support flexible compression ratios, FlexTok Bachmann et al. (2025), One-D-Piece Miwa et al. (2025), VibeToken Patel et al. (2026) and ELIT Haji-Ali et al. (2026) employ tail-drop training, while ALIT Duggal et al. (2024) introduces recurrent allocation. Meanwhile, GaussianToken Dong et al. (2025) and GPSToken Zhang et al. (2025) leverage parametric 2D Gaussians to dynamically model the shape and position of different image regions.

Although efficient, 1D tokenizers often lose fine-grained spatial structure, and the interplay between dynamic token number and adaptive location remains underexplored. In particular, when only a few tokens are allocated, these limited yet valuable tokens should focus on the most informative regions. Motivated by these limitations, we propose a hybrid 1D and 2D tokenizer that injects 2D spatial cues into the 1D token representation through a fusion layer.

Image Generation. Image generation serves as one of the most representative applications of image tokenization, where the learned latent representations are used to synthesize high-fidelity visual content. Early approaches performed generation directly in pixel space Kingma & Welling (2013); Van Den Oord et al. (2017); Razavi et al. (2019); Goodfellow et al. (2014); Ho et al. (2020), but exhibited limited performance. LDM Rombach et al. (2022) demonstrated that training diffusion models in the latent space of a 2D VAE Kingma & Welling (2013) yields substantial improvements. Subsequent works further improve network architectures Bao et al. (2023); Peebles & Xie (2023); Yao et al. (2025); Wang et al. (2026) and training strategies Gao et al. (2023); Ma et al. (2024); Lipman et al. (2022); Yu et al. (2024b); Wu et al. (2025); Zheng et al. (2026). Another line of research adopts vector-quantized tokenizers Van Den Oord et al. (2017); Lee et al. (2022); Esser et al. (2021) and autoregressive models Esser et al. (2021); Yu et al. (2023); Sun et al. (2024); Tian et al. (2024); Han et al. (2025) for image synthesis. In this work, we focus on continuous tokenizers and diffusion models for image generation.

3 METHODOLOGY

In this section, we first introduce the partition algorithm, which uses a heatmap derived from the gradient map to allocate tokens and divide images into non-uniform regions. We then describe the architecture and training paradigm of DynamicTok. Finally, we explain how DynamicTok integrates with existing generators and enables adaptive-length generation via a lightweight structure generator.

3.1 PARTITION ALGORITHM

By default, we use a gradient map to measure image complexity, and further evaluate saliency maps as an alternative in the ablation study. As illustrated in Fig. 2, we first convert the image to a gray-scale image and apply a Sobel operator Sobel et al. (1968) to compute the gradient map of each image, where regions with richer textures yield higher gradient values, indicating greater reconstruction difficulty. The mean gradient value of the image is used to quantify its overall complexity. To determine the number of tokens allocated to each image, we statistically analyze the distribution of mean gradient values across the entire dataset. Let the quantile of the mean gradient value of the current image be denoted by $q \in [0, 1]$. The token count is determined by a simple linear function:

$$N = N_L + \lfloor (N_H - N_L + 4) \cdot q \,/\, 4 \rfloor \cdot 4, \tag{1}$$

where N_L and N_H represent the minimum and maximum token budgets, respectively. For example, when $[N_L, N_H] = [32, 96]$, the assigned token count varies between $\{32, 36, \ldots, 96\}$ according to the mean gradient value of the image.

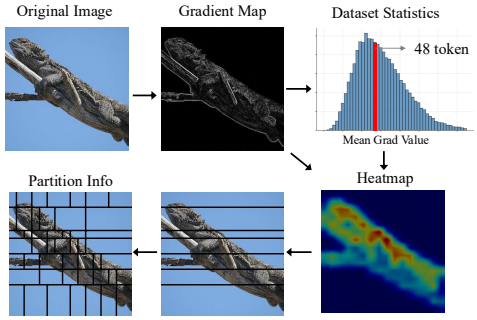


Figure 2: Partition Process. We use Sobel operator Sobel et al. (1968) to extract the image gradient map. The token count is determined by the dataset's mean gradient quantile, assigning more tokens to high-gradient images. The gradient map is pooled and normalized into a heatmap, which is split row- and column-wise so each region has an approximately equal cumulative value.

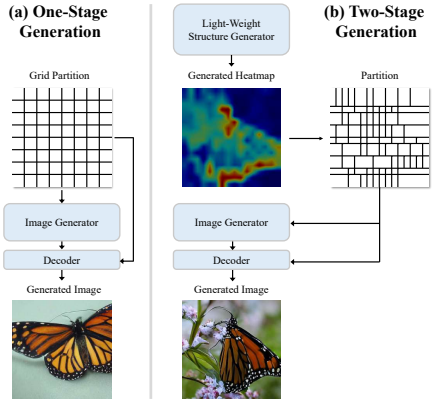


Figure 3: (a) DynamicTok integrates seamlessly with existing generators by adopting grid partitioning during generative training and inference. (b) For adaptive-length generation, a lightweight generator first predicts the heatmap.

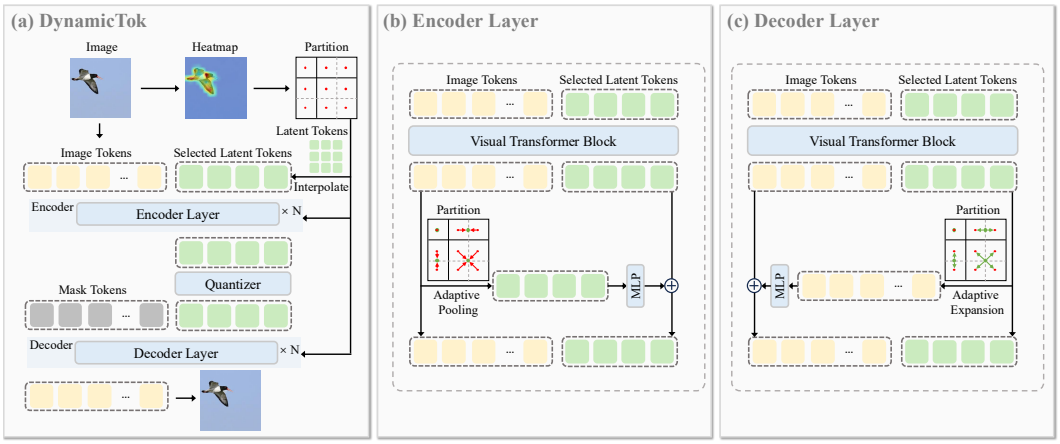


Figure 4: (a) Overall architecture of DynamicTok. We first extract a heatmap from the original image, which is then partitioned into regions of equal cumulative values. (b) Encoder Layer. Image tokens and latent tokens are processed by a visual transformer block with full attention, after which the image tokens are adaptively pooled according to partition information and fused with the latent tokens. (c) Decoder Layer. Symmetric to the encoder, latent tokens and masked tokens are input into a visual transformer block, then the latent tokens are adaptively expanded according to partition information and fused with image tokens.

To partition each image according to local complexity, we pool and normalize the gradient map into a 16×16 heatmap using average pooling followed by softmax normalization. To obtain regions with approximately equal cumulative values while supporting parallel processing, we adopt a row-column splitting strategy. As shown in Fig. 2, the heatmap is first split horizontally into rows with equal cumulative values, and each row is then divided vertically into rectangular regions with similar cumulative sums.

3.2 DynamicTok

We employ the partition algorithm described in Sec. 3.1 to achieve spatial and quantitative adaptivity. The overall architecture of DynamicTok is illustrated in Fig. 4(a). DynamicTok builds on a 1D tokenizer Yu et al. (2024a) and additionally inject local feature to each latent token. To initialize the latent tokens, we first compute the center of each partitioned region and obtain the set of positions $\text{Pos} = \{p_1, p_2, \ldots, p_N\}$. We then interpolate the learnable grid-structure latent tokens Z_G at these

positions to produce the selected latent tokens:

$$Z = \text{Interp}(Z_G, \text{Pos}), \quad Z \in \mathbb{R}^{N\times D}, Z_G \in \mathbb{R}^{M\times M\times D}. \tag{2}$$

The input image is patchified into $P_I \in \mathbb{R}^{h\times w\times D}$, which is then concatenated with the latent tokens and passed into the visual encoder.

Encoder. The encoder consists of multiple identical encoder layers, which are shown in Fig. 4(b). Both image tokens and latent tokens are first fed into a visual transformer block Dosovitskiy (2020) with full attention. Next, guided by partition information, the image tokens within the same region are aggregated into a single token by average pooling, resulting in pooled representations $\text{Pool}(P) \in \mathbb{R}^{N\times D}$ that share the same shape as Z. We then fuse $\text{Pool}(P)$ with the latent tokens Z at each spatial position to inject adaptive local features into the latent representations. This design enables each latent token to capture both global context and locally adaptive information. By default, we adopt additive fusion with an MLP. The latent token is updated as:

$$Z \leftarrow Z + \text{MLP}(Z + \text{Pool}(P)). \tag{3}$$

Quantizer. The latent tokens produced by the encoder share the same high-dimensional embedding space as the visual transformer block. To reduce dimensionality and impose an information bottleneck, we insert a quantizer between the encoder and decoder. The quantizer projects the latent tokens into a compact space $z \in \mathbb{R}^{N\times d}(d \ll D)$ which serves for both image representation and generation. Before being passed to the decoder, z is projected back to the original embedding dimension, yielding $\hat{Z} \in \mathbb{R}^{N\times c}$ in the original embedding dimension. By default, we adopt the soft quantization mechanism from SoftVQ Chen et al. (2025b), which enables continuous token projection.

Decoder. The decoder takes the latent tokens from the quantizer and the learnable masked tokens $P_M \in \mathbb{R}^{h\times w\times D}$. As shown in Fig. 4(c), the decoder layer is symmetric to the encoder layer with a visual transformer block and a fusion stage. During fusion, the latent tokens are expanded according to the partition information to match the spatial dimensions of the image tokens, where all masked tokens within the same region share the same latent token. The expanded latent features $\text{Expand}(\hat{Z}) \in \mathbb{R}^{h\times w\times D}$ are then fused with the masked tokens through another MLP:

$$P_M \leftarrow P_M + \text{MLP}(P_M + \text{Expand}(\hat{Z})). \tag{4}$$

Training Objective. Following SoftVQ-VAE Chen et al. (2025b), DynamicTok is trained with a combination of reconstruction, perceptual, adversarial, KL-divergence, and alignment losses. The alignment loss maximizes the pairwise similarity between expanded latent tokens output by the last encoder layer and the feature maps of a pretrained visual foundation model. Given the pretrained feature $y \in \mathbb{R}^{h\times w\times D}$, the alignment loss is as follows:

$$\mathcal{L}_{align} = -\frac{1}{hw}\sum_{i=1}^{h}\sum_{j=1}^{w}\text{sim}(y^{[i,j]}, h_\phi(\text{Expand}(Z))^{[i,j]}), \tag{5}$$

where h_ϕ is used to project latent tokens to the same dimension as pretrained feature. The overall loss is as follows:

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \lambda_1\mathcal{L}_{\text{percep}} + \lambda_2\mathcal{L}_{\text{adv}} + \lambda_3\mathcal{L}_{\text{align}} + \lambda_4\mathcal{L}_{\text{KL}}. \tag{6}$$

3.3 Dynamic Generator

Unlike Gaussian-Parameterized Tokenizers Dong et al. (2025); Zhang et al. (2025), DynamicTok seamlessly integrates with existing generative models by using a grid partition to extract latent tokens. As shown in Fig. 3(a), the partition remains consistent across all images and is utilized only by the decoder during image synthesis.

To further support generation with dynamic token counts, we adopt a two-stage strategy. As shown in Fig. 3(b), a lightweight structure generator first predicts the heatmap, which is used to derive partition information fed into both the image generator and decoder. However, we find that directly predicting a low-resolution heatmap leads to suboptimal results. Instead, we first generate a gray-scale image at the original resolution, convert it to a gradient map, and then downsample & normalize it into a

Table 1: Reconstruction results on ImageNet 256×256. "Disc." and "Cont." denote "Discrete" and "Continuous," respectively. The average token counts for $[32, 96]$ and $[64, 192]$ is 64 and 128. DynamicTok surpass approaches with same token budgets.

Tokenizer	Structure	Latent	Tokens	rFID↓
TiTok-B Yu et al. (2024a)	1D	Disc.	64	1.70
FlexTok Bachmann et al. (2025)	1D	Disc.	64	1.37
SoftVQ Chen et al. (2025b)	1D	Cont.	64	0.88
TiTok-S Yu et al. (2024a)	1D	Disc.	128	1.71
MAETok Chen et al. (2025a)	1D	Cont.	128	0.48
DCAE Chen et al. (2025c)	2D	Cont.	64	0.69
GPSTok Zhang et al. (2025)	2D	Cont.	128	0.65
VQGAN Esser et al. (2021)	2D	Disc.	256	7.94
VA-VAE Yao et al. (2025)	2D	Cont.	256	0.28
l-DeTok Yang et al. (2026)	2D	Cont.	256	0.68
RAE Zheng et al. (2026)	2D	Cont.	256	0.49
SD-VAE Rombach et al. (2022)	2D	Cont.	1024	0.62
DynamicTok-B	Hybrid	Cont.	[32, 96]	0.65
DynamicTok-L	Hybrid	Cont.	[32, 96]	0.57
DynamicTok-B	Hybrid	Cont.	[64, 192]	0.39

heatmap. Because the gradient map is heavily downsampled, the gray-scale image prediction need not be precise, allowing the first-stage tokenizer and generator to remain lightweight.

We make only minimal modifications to the image generator, replacing fixed positional embeddings with interpolated positional embeddings determined by the spatial locations of the latent tokens. Each latent token is positioned at the center of its corresponding region. To support flexible token-budget control, we randomly sample the number of tokens during two-stage training.

4 Experiments

4.1 Experimental Settings

Implementation Details. We build DynamicTok upon the SoftVQ-VAE codebase Chen et al. (2025b). Following SoftVQ-VAE, we use pretrained DINO-v2-Large Oquab et al. (2023) for feature alignment in Eq. 5 and DINO-S as the discriminator. By default, we set latent channel dimension $d = 32$, token range $[N_L, N_H] = [32, 96]$ and $M = h = w = 16$. Each fusion layer is implemented as a two-layer MLP and inserted after every Transformer block except the final one. Following SoftVQ-VAE Chen et al. (2025b) and REG Wu et al. (2025), we set the loss weights to $\lambda_1 = 1.0$, $\lambda_2 = 0.2$, $\lambda_3 = 0.1$, and $\lambda_4 = 0.01$. We adopt SiT-XL Ma et al. (2024) as the image generator. For grayscale-image prediction, we use ViT-S as the tokenizer and LightningDiT-S Yao et al. (2025) as the structure generator.

Evaluation. All experiments are conducted on ImageNet-1K Deng et al. (2009), with images resized and center-cropped to 256 × 256. We evaluate reconstruction quality using reconstruction Fréchet Inception Distance (rFID). For generative performance, we report generation FID (gFID) and Inception Score (IS), both with and without classifier-free guidance (CFG) Ho & Salimans (2022).

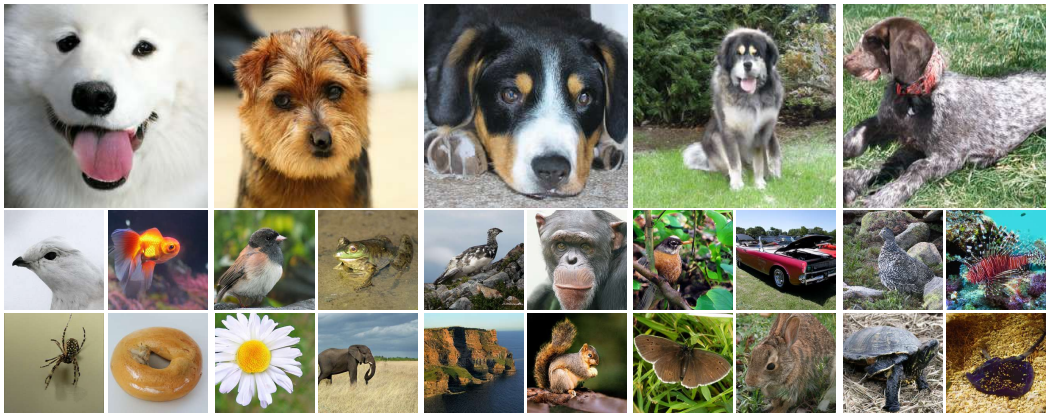
4.2 Main Results

We compare DynamicTok against other SOTA tokenizer and generator on ImageNet 256×256.

Reconstruction. As shown in Tab. 1, DynamicTok achieves state-of-the-art reconstruction performance with average 64 and 128 tokens, reaching rFID of 0.57 and 0.39, respectively. DynamicTok outperforms many tokenizers that employ 256 or more tokens while using substantially fewer tokens, demonstrating its efficiency. Beyond its strong reconstruction performance, DynamicTok allows flexible adjustment of token quantity and spatial allocation. As shown in the left of Fig. 6, fewer tokens can be assigned to simpler images to reduce redundancy, while more tokens can be allocated to complex one for finer reconstruction. Notably, although trained on a token range of $[32, 96]$, DynamicTok generalizes beyond this range. Increasing the token count to 256 further improves

Table 2: System-level comparison on ImageNet 256×256 for class-conditional generation.

Tokenizer	Tokens	Gen. Models	Gen. (w/o CFG)		Gen. (w/ CFG)	
			gFID↓	IS↑	gFID↓	IS↑
2D Tokenization						
VQGAN Esser et al. (2021)	256	Taming-Trans Esser et al. (2021)	5.20	290.3	-	-
VQGAN Esser et al. (2021)	576	LlamaGen Sun et al. (2024)	-	-	2.18	263.3
MSRQ Tian et al. (2024)	680	VAR Tian et al. (2024)	-	-	1.92	323.1
SD-VAE(KL-16) Rombach et al. (2022)	256	MAR Li et al. (2024)	2.35	227.8	1.55	303.7
		DiT-XL/2 Peebles & Xie (2023)	9.62	121.5	2.27	278.2
SD-VAE Rombach et al. (2022)	1024	SiT-XL/2 Ma et al. (2024)	8.30	131.7	2.06	270.3
		REPA Yu et al. (2024b)	5.90	157.8	1.42	305.7
VAVAE Yao et al. (2025)	256	LightningDiT Yao et al. (2025)	2.17	205.6	1.35	295.3
GPSToken Zhang et al. (2025)	128	SiT-XL/1 Ma et al. (2024)	-	-	1.50	-
l-DeTok Yang et al. (2026)	256	MAR Li et al. (2024)	1.86	238.6	1.35	304.1
RAE Zheng et al. (2026)	256	DiT^{DH}-XL Zheng et al. (2026)	1.51	242.9	1.13	262.6
1D Tokenization						
TiTok-B Yu et al. (2024a)	64	MaskGIT Chang et al. (2022)	3.08	192.5	2.48	214.7
TiTok-S Yu et al. (2024a)	128	MaskGIT Chang et al. (2022)	4.44	168.2	1.97	281.8
FlexTok Bachmann et al. (2025)	64	LlamaGen Sun et al. (2024)	-	-	1.76	286.4
MAETok Chen et al. (2025a)	128	LightningDiT Yao et al. (2025)	2.21	208.3	1.73	308.4
		SiT-XL/1 Ma et al. (2024)	2.31	216.5	1.67	311.2
SoftVQ Chen et al. (2025b)	64	DiT-XL/1 Peebles & Xie (2023)	6.62	129.2	3.29	262.5
		SiT-XL/1 Ma et al. (2024)	5.98	138.0	1.78	279.0
One-D-Piece Miwa et al. (2025)	256	MaskGIT Chang et al. (2022)	-	-	2.70	259.3
VibeToken Patel et al. (2026)	64	VibeTokenGen Patel et al. (2026)	-	-	3.62	226.93
Ours						
DynamicTok-B (1-stage)	64	SiT-XL/1	2.32	219.0	1.40	323.5
DynamicTok-B (2-stage)	[32, 96]	SiT-XL/1	2.57	218.1	1.63	327.3
DynamicTok-L (1-stage)	64	SiT-XL/1	2.48	217.9	1.56	303.1



32 Tokens 48 Tokens 64 Tokens 80 Tokens 96 Tokens

Figure 5: Generation results with different token counts using two-stage generation. Fewer tokens produce simpler images, while more tokens yield images with richer structure and complex texture. Notably, even 32 tokens are sufficient to generate high-quality images.

reconstruction quality; for example, in the third row of the left image in Fig. 6, using 256 tokens produces image with higher fidelity. As illustrated in the right panel of Fig. 6, users can also manually emphasize specific regions, such as assigning higher weights to faces for enhanced local detail.

Generation. DynamicTok also supports high-quality image synthesis. As shown in Tab. 2, when paired with SiT-XL, it achieves the gFID of 1.40, setting new records among generators using fewer than 256 tokens and remaining competitive with the latest methods. Although two-stage generation incurs a performance degradation, it still outperforms strong baselines such as MAETok Chen et al. (2025a). Moreover, as demonstrated in the ablation study, it enables flexible token-budget allocation, improving efficiency with fewer tokens and supporting structural control.

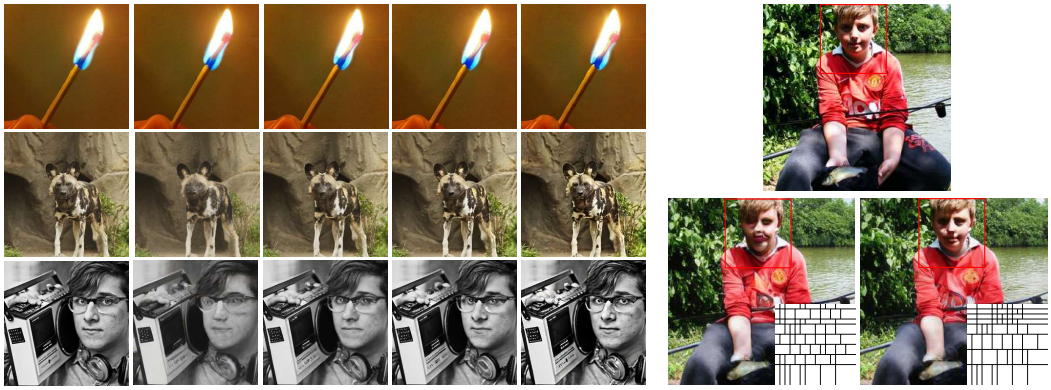


Figure 6: Left: Reconstruction with varying token counts. Simple images can be reconstructed with as few as 32 tokens, while complex images benefit from more tokens for higher fidelity. Although trained with 32–96 tokens, DynamicTok can use larger token budgets at inference to enhance quality. Right: DynamicTok also allows manual adjustment of token allocation across local regions.

Table 3: Comparison of different dynamic strategies in training and inference. "–" denotes the fixed-grid partition, while "Grad" indicates gradient-map-based partition. Both token quantity and spatial adaptivity improve performance.

Training		Inference		rFID↓
Num	Map	Num	Map	
64	-	64	-	0.96
64	Grad	64	-	0.75
64	Grad	64	Grad	0.71
[32, 96]	Grad	64	-	0.69
[32, 96]	Grad	[32, 96]	Grad	0.65

Table 4: Ablation of different design of DynamicTok. Here Training and Inference use same token counts.

Model	Map	Num	Partition	rFID↓
ViT-B	Gradient	[32, 96]	Row	0.65
ViT-B	Gradient	[16, 112]	Row	0.71
ViT-B	Gradient	[64, 192]	Row	0.39
ViT-L	Gradient	[32, 96]	Row	0.57
ViT-B	Gradient	[32, 96]	Iter	0.67
ViT-B	Saliency	[32, 96]	Row	0.67
ViT-B	Saliency	[32, 96]	Iter	0.65

4.3 Ablation Studies

Effect of Quantity and Spatial Adaptivity. We demonstrate that training tokenizer with adaptive token numbers and positions enhances both reconstruction and generation performance. As shown in Tab. 3, tokenizer trained with a fixed token count and a grid partition, i.e., the uniform baseline, achieve the lowest performance. Introducing adaptive spatial partitioning during both training and inference improves the rFID from 0.96 to 0.71. Further incorporating an adaptive token quantity reduces the rFID to 0.65. We also find that dynamic training improves fixed inference. When trained with dynamic token quantities and positions, the model reduces rFID from 0.96 to 0.69 under fixed-number, fixed-position inference. Dynamic training also improves generation performance: as shown in Tab. 6, the uniform baseline underperforms DynamicTok in terms of generation quality.

Flexibility of two-stage generation. Two-stage generation enables explicit control over computational efficiency and spatial structure at inference time. As shown in Fig. 5, DynamicTok generates high-quality images across varying token budgets. Tab. 7 compares the efficiency of DynamicTok with that of existing methods. We measure generation latency with a batch size of 256 on a single GPU. Owing to its two-stage design, DynamicTok supports flexible token-budget allocation based on computational constraints. With 48 tokens, DynamicTok outperforms SoftVQ-VAE Chen et al. (2025b) in both generation quality and efficiency. Reducing the budget to 32 tokens further lowers inference latency while maintaining generation quality comparable to SiT Ma et al. (2024), making DynamicTok suitable for resource-constrained settings. As shown in Fig. 7, the two-stage design also enables explicit structural control. Specifically, a heatmap extracted from a reference image can condition the generation of a new image, enabling layout transfer.

Design of Tokenizer. We ablate several design choices of DynamicTok, including model size, token range, heatmap type, and partitioning strategy. As shown in Tab. 4, dynamic token allocation improves

Table 5: Generation quality and time occupancy across heatmap prediction strategies.

Heatmap	gFID w/ CFG ↓	Time Occ ↓
-	1.40	-
Real	1.37	-
Lo-Res	3.54	4.1 %
Hi-Res	1.63	4.4 %

Table 6: Comparison between DynamicTok and uniform baseline in one-stage generation.

Tokenizer	gFID ↓ (w/o CFG)	gFID ↓ (w/ CFG)
Dynamic	2.32	1.40
Uniform	3.69	1.66

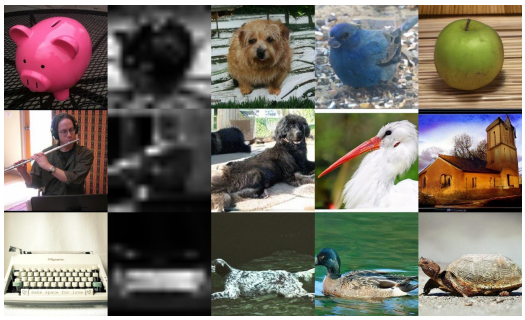


Figure 7: Original images (first column), heatmaps (second column), and heatmap-guided structure-transfer results (remaining columns).

Table 7: Comparison of generation FLOPs and latency across methods. All methods use SiT-XL as the generator. Here, T denotes TFLOPs. For DynamicTok, Gen. FLOPs and Gen. Latency include the costs of both first- and second-stage generation.

Method	Tokens	Gen. FLOPs	Gen. Latency	gFID (w/ CFG)
DynamicTok (2-stage)	[32, 96]	32.3 G	217.1 ms	1.63
DynamicTok (2-stage)	64	30.9 G	153.6 ms	1.64
DynamicTok (2-stage)	48	23.6 G	108.4 ms	1.69
DynamicTok (2-stage)	32	15.4 G	83.0 ms	2.18
SoftVQ-VAE Chen et al. (2025b)	64	29.1 G	143.6 ms	1.78
MAETok Chen et al. (2025a)	128	58.4 G	291.5 ms	1.67
SD-VAE Rombach et al. (2022)	1024	118.7 G	799.7 ms	2.06

flexibility, whereas an overly broad token range degrades performance because small budgets (e.g., 16 tokens) are insufficient for accurate reconstruction. We therefore use $[N_L, N_H] = [32, 96]$ by default. Scaling the backbone from ViT-B to ViT-L improves rFID from 0.65 to 0.57. We also evaluate iterative row–column partitioning and saliency-based heatmaps Liu et al. (2021), with visualizations provided in the Appendix. The last three rows of Tab. 4 show comparable rFID across these variants, demonstrating robustness to tokenizer design choices. However, saliency maps perform worse in two-stage generation (see Appendix); therefore, we adopt gradient maps by default.

Heatmap Prediction. We compare different first-stage heatmap prediction strategies in Tab. 5. "–" denotes one-stage generation; "Real" uses training-set heatmaps; "Hi-Res" and "Lo-Res" indicate first-stage prediction of a grayscale image and a low-resolution heatmap, respectively. Predicting a grayscale image achieves much better gFID than directly predicting a low-resolution heatmap. Compared with low-resolution prediction, the high-resolution variant only adds a lightweight ViT-Small decoder with negligible overhead, while producing substantially better results. In both settings, the first stage takes less than 5% of the total time. We also compare with real heatmaps sampled from training images, which perform better than both one-stage and two-stage generation, indicating that heatmap generation may introduce minor errors. Visualizations of generated heatmap and corresponding images are provided in the Appendix.

5 Conclusion

We present DynamicTok, an image tokenizer that enables dynamic control over both token quantity and spatial allocation. It adaptively assigns fewer tokens to visually simple images and regions to reduce redundancy and more tokens to complex ones to enhance reconstruction accuracy, while also allowing manual adjustment of token distribution when needed. DynamicTok outperforms approaches with the same token budgets in image reconstruction. DynamicTok makes two primary contributions to image generation. First, when integrated with a fixed-token generator, it achieves state-of-the-art generation quality among methods using fewer than 256 tokens. Second, when incorporated into our proposed two-stage generation framework, it enables flexible control over token quantity and spatial allocation while maintaining competitive generation quality.

AI USE STATEMENT

In this work, we used generative AI tools for design or provide feedback on research methodology or experiments, implement methods, assist with translation, clean and reformat dataset, support qualitative and thematic data analysis, interpret results. We have not used generative AI tools for Generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses. Additionally, we used generative AI tools for create or edit software code, creation of artifacts, draft parts of a research paper, transcribe recordings of research material, summarize or analyse existing literature, brainstorming, sourcing/searching for information, edit a research paper to improve readability, identify relevant literature, format references, suggest a structure for a research paper. We have reviewed all AI-assisted work. we checked LLM-generated research ideas for potential plagiarism through a manual literature survey and LLM-generated code was verified and tested for correctness. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies adaptive image tokenization and generation using the publicly available ImageNet-1K benchmark. We do not collect new data, conduct studies involving human participants, or release personally identifiable information. As with other generative image models, the proposed method could potentially be misused to synthesize misleading, biased, or otherwise harmful visual content. We therefore recommend that downstream users apply appropriate safeguards, conduct bias and safety evaluations, and comply with relevant laws and platform policies before deployment. The method is intended for research on efficient visual representation learning and image generation; the authors are not aware of conflicts of interest that would compromise this work and take responsibility for its research integrity and reported results.

REPRODUCIBILITY STATEMENT

We support reproducibility by describing the DynamicTok architecture, adaptive partition algorithm, training objective, and generation procedure in Sections 3 and 4. Section 4 specifies the experimental setup, including the dataset, image preprocessing, model configurations, token-budget settings, baseline comparisons, and evaluation metrics. The appendix provides further architectural details, position-embedding formulations, ablations, and qualitative results. Upon acceptance, we will release the implementation code.

REFERENCES

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Roman Bachmann, Jesse Allardice, David Mizrahi, Enrico Fini, Oğuzhan Fatih Kar, Elmira Amirloo, Alaaeldin El-Nouby, Amir Zamir, and Afshin Dehghan. Flextok: Resampling images into 1d token sequences of flexible length. In *Forty-second International Conference on Machine Learning*, 2025.

Fan Bao, Shen Nie, Kaiwen Xue, Yue Cao, Chongxuan Li, Hang Su, and Jun Zhu. All are worth words: A vit backbone for diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 22669–22679, 2023.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11315–11325, 2022.

Hao Chen, Yujin Han, Fangyi Chen, Xiang Li, Yidong Wang, Jindong Wang, Ze Wang, Zicheng Liu, Difan Zou, and Bhiksha Raj. Masked autoencoders are effective tokenizers for diffusion models. In *Forty-second International Conference on Machine Learning*, 2025a.

Hao Chen, Ze Wang, Xiang Li, Ximeng Sun, Fangyi Chen, Jiang Liu, Jindong Wang, Bhiksha Raj, Zicheng Liu, and Emad Barsoum. Softvq-vae: Efficient 1-dimensional continuous tokenizer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 28358–28370, 2025b.

Junyu Chen, Han Cai, Junsong Chen, Enze Xie, Shang Yang, Haotian Tang, Muyang Li, and Song Han. Deep compression autoencoder for efficient high-resolution diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2025c.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Jiajun Dong, Chengkun Wang, Wenzhao Zheng, Lei Chen, Jiwen Lu, and Yansong Tang. Gaussiantoken: An effective image tokenizer with 2d gaussian splatting. *arXiv preprint arXiv:2501.15619*, 2025.

Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Tianqi Du, Yifei Wang, and Yisen Wang. On the role of discrete tokenization in visual representation learning. In *The Twelfth International Conference on Learning Representations*, 2024.

Shivam Duggal, Phillip Isola, Antonio Torralba, and William T Freeman. Adaptive length image tokenization via recurrent allocation. In *First Workshop on Scalable Optimization for Efficient and Adaptive Foundation Models*, 2024.

Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12873–12883, 2021.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

Shanghua Gao, Pan Zhou, Ming-Ming Cheng, and Shuicheng Yan. Mdtv2: Masked diffusion transformer is a strong image synthesizer. *arXiv preprint arXiv:2303.14389*, 2023.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Moayed Haji-Ali, Willi Menapace, Ivan Skorokhodov, Dogyun Park, Anil Kag, Michael Vasilkovsky, Sergey Tulyakov, Vicente Ordonez, and Aliaksandr Siarohin. One model, many budgets: Elastic latent interfaces for diffusion transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4558–4568, 2026.

Jian Han, Jinlai Liu, Yi Jiang, Bin Yan, Yuqi Zhang, Zehuan Yuan, Bingyue Peng, and Xiaobing Liu. Infinity$[0.5, \infty)$: Scaling bitwise autoregressive modeling for high-resolution image synthesis. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15733–15744. IEEE, 2025.

D Hendrycks. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Mengqi Huang, Zhendong Mao, Zhuowei Chen, and Yongdong Zhang. Towards accurate image coding: Improved autoregressive image generation with dynamic vector quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22596–22605, 2023.

Weinan Jia, Mengqi Huang, Nan Chen, Lei Zhang, and Zhendong Mao. Dˆ 2it: Dynamic diffusion transformer for accurate image generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 12860–12870, 2025.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain, et al. Matryoshka representation learning. *Advances in Neural Information Processing Systems*, 35:30233–30249, 2022.

Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11523–11532, 2022.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pp. 19730–19742. PMLR, 2023.

Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *Advances in Neural Information Processing Systems*, 37: 56424–56445, 2024.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Nian Liu, Ni Zhang, Kaiyuan Wan, Ling Shao, and Junwei Han. Visual saliency transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4722–4732, 2021.

Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, pp. 23–40. Springer, 2024.

Keita Miwa, Kento Sasaki, Hidehisa Arai, Tsubasa Takahashi, and Yu Yamaguchi. One-d-piece: Image tokenizer meets quality-controllable compression. *arXiv preprint arXiv:2501.10064*, 2025.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Maitreya Patel, Jingtao Li, Weiming Zhuang, Yezhou Yang, and Lingjuan Lv. Vibetoken: Scaling 1d image tokenizers and autoregressive models for dynamic resolution generations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2058–2068, 2026.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4195–4205, 2023.

Liao Qu, Huichao Zhang, Yiheng Liu, Xu Wang, Yi Jiang, Yiming Gao, Hu Ye, Daniel K Du, Zehuan Yuan, and Xinglong Wu. Tokenflow: Unified image tokenizer for multimodal understanding and generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 2545–2555, 2025.

Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 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.

Irwin Sobel, Gary Feldman, et al. A 3x3 isotropic gradient operator for image processing. *a talk at the Stanford Artificial Project in*, 1968:271–272, 1968.

Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*, 2024.

Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *Advances in neural information processing systems*, 37:84839–84865, 2024.

Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

Shuai Wang, Zhi Tian, Weilin Huang, and Limin Wang. Ddt: Decoupled diffusion transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 40633–40642, 2026.

Ge Wu, Shen Zhang, Ruijing Shi, Shanghua Gao, Zhenyuan Chen, Lei Wang, Zhaowei Chen, Hongcheng Gao, Yao Tang, Jian Yang, et al. Representation entanglement for generation: Training diffusion transformers is much easier than you think. *arXiv preprint arXiv:2507.01467*, 2025.

Wilson Yan, Volodymyr Mnih, Aleksandra Faust, Matei Zaharia, Pieter Abbeel, and Hao Liu. Elastictok: Adaptive tokenization for image and video. In *The Thirteenth International Conference on Learning Representations*, 2025.

Jiawei Yang, Tianhong Li, Lijie Fan, Yonglong Tian, and Yue Wang. Latent denoising makes good tokenizers. In *The Fourteenth International Conference on Learning Representations*, 2026.

Jingfeng Yao, Bin Yang, and Xinggang Wang. Reconstruction vs. generation: Taming optimization dilemma in latent diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 15703–15712, 2025.

Lijun Yu, José Lezama, Nitesh B Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Vighnesh Birodkar, Agrim Gupta, Xiuye Gu, et al. Language model beats diffusion–tokenizer is key to visual generation. *arXiv preprint arXiv:2310.05737*, 2023.

Qihang Yu, Mark Weber, Xueqing Deng, Xiaohui Shen, Daniel Cremers, and Liang-Chieh Chen. An image is worth 32 tokens for reconstruction and generation. *Advances in Neural Information Processing Systems*, 37:128940–128966, 2024a.

Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. *arXiv preprint arXiv:2410.06940*, 2024b.

Zhengqiang Zhang, Rongyuan Wu, Lingchen Sun, and Lei Zhang. Gpstoken: Gaussian parameterized spatially-adaptive tokenization for image representation and generation. *arXiv preprint arXiv:2509.01109*, 2025.

Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders. In *International Conference on Learning Representations*, volume 2026, pp. 35791–35820, 2026.

A Model Architectures

Position Embedding For DynamicTok, we derive position embeddings for dynamic tokens by interpolating the frozen absolute position embeddings of the pretrained DINO-V2 encoder Oquab et al. (2023), denoted as $\text{PE}_{dino} \in \mathbb{R}^{M \times M \times D}$. Given the 2D spatial coordinates of all tokens:

$$\text{Pos} = \{p_1, \ldots, p_N\} = \{(x_1, y_1), \ldots, (x_N, y_N)\}, \tag{7}$$

we apply bilinear interpolation to obtain the position embedding for each token:

$$\text{PE}(p_i) = \text{Interp}(\text{PE}_{dino}, p_i). \tag{8}$$

For one-stage generator, we use 2D sinusoidal position embeddings following SiT Ma et al. (2024):

$$\begin{aligned}
&\text{PE}(x, 2i) = \sin\left(\frac{x}{10000^{2i/(D/2)}}\right), \\
&\text{PE}(x, 2i+1) = \cos\left(\frac{x}{10000^{2i/(D/2)}}\right), \\
&\text{PE}(y, 2i) = \sin\left(\frac{y}{10000^{2i/(D/2)}}\right), \\
&\text{PE}(y, 2i+1) = \cos\left(\frac{y}{10000^{2i/(D/2)}}\right), \\
&\text{PE}(p_i) = \text{PE}(x, y) = \text{concat}\left[\text{PE}(x), \text{PE}(y)\right], \\
&x, y \in [0, 1, \ldots, N-1]
\end{aligned} \tag{9}$$

In the two-stage generation framework, the dynamic position embeddings follow the same form as Eq. 9, but the spatial coordinates (x, y) are extended from discrete grid indices to continuous values in the range $[0, , N-1]$.

DynamicTok. The transformers in DynamicTok follow the DINO-v2 architecture Oquab et al. (2023). The encoder Transformer blocks are initialized with pretrained DINO-v2 weights, whereas the decoder Transformer blocks are trained from scratch following SoftVQ-VAE Chen et al. (2025b). All fusion layers are implemented as two-layer MLPs with GELU activations Hendrycks (2016) and Layer Normalization Ba et al. (2016). Fusion layers are inserted after every encoder and decoder block except the final block.

B Ablation of fusion layers

We ablate the fusion layers in the encoder and decoder. As shown in Tab. 8, removing all fusion layers limits the latent tokens' ability to capture regional information, resulting in the worst reconstruction performance. Introducing fusion layers into the encoder improves rFID from 1.36 to 0.81, while incorporating them into both the encoder and decoder achieves the best rFID of 0.65. We attribute these improvements to the effectiveness of fusion layers in enhancing the spatial awareness of latent tokens.

Table 8: Ablation of fusion layers.

Fusion Layer in Encoder	Fusion Layer in Decoder	rFID ↓
		1.36
✓		0.81
✓	✓	0.65

C Two-Stage Generation with Saliency Map

We use VST (Visual Saliency Transformer) Liu et al. (2021) to predict saliency maps (visualized in Fig. 8) which capture foreground objects in images. We predict a downsampled 16×16 saliency map in the first stage, as low-resolution saliency maps exhibit clearer structural patterns than low-resolution gradient maps. As shown in Tab. 9, the performance gap between ground-truth and predicted heatmaps is substantially larger for saliency maps (1.43 vs. 0.16). We attribute this degradation to the low recall of predicted saliency maps, as they often consist of a single dominant region with limited spatial distinctiveness.

D More Visualization

Visualization of different heatmap and partition strategy. The visualization of row-column partition and saliency map is provided in Fig. 8.

Visualization of two-stage generation. Fig. 9 shows the grayscale images ($256 \times 256 \times 1$) predicted in the first stage and the corresponding color images ($256 \times 256 \times 3$) generated in the second

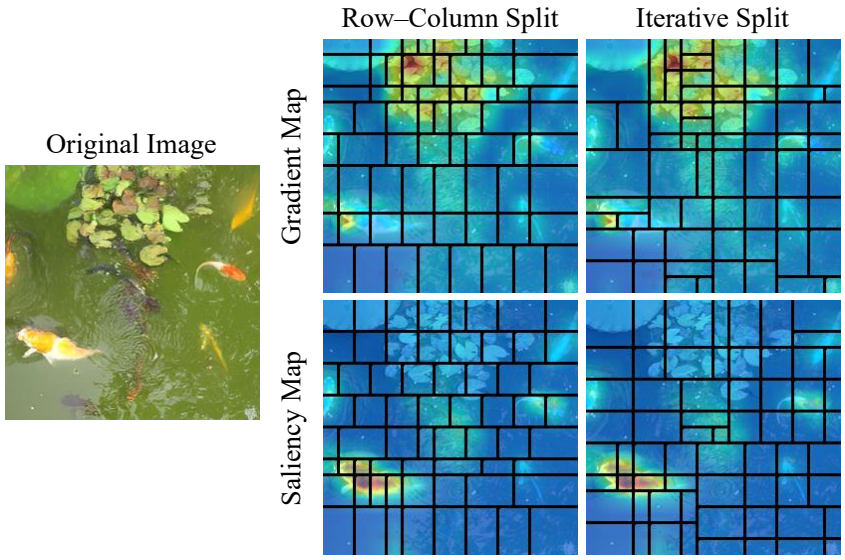


Figure 8: Different heatmap generation and partition algorithms.

Table 9: Quantitative comparison of using gradient and saliency map for two-stage generation without CFG. "Real" denotes using heatmap sampled from training set.

Map	Source	gFID (w/o CFG)	Precision	Recall
Gradient	Real	2.41	0.78	0.61
Gradient	Pred	2.57	0.79	0.62
Saliency	Real	3.34	0.77	0.62
Saliency	Pred	4.77	0.86	0.46

stage. Although the grayscale predictions are often low-quality, they provide sufficient structural guidance for the second stage. Because the second stage uses only weak structural constraints (the partition), the final images may slightly deviate from the grayscale inputs (e.g., row 5). The predicted grayscale image can also be abnormal (row 6) which may influence final results and cause performance degration.

Visualization of one-stage generation. Uncurated samples from our one-stage generator are displayed in Figs. 10–19. We use CFG=3.25 for all generated images.

E Limitation and Future Work

Although DynamicTok improves reconstruction performance through dynamic token assignment and supports variable-length tokens during generation via a two-stage pipeline, its two-stage generation performance still lags behind one-stage generation. Future work may explore more effective heatmap generation methods, such as end-to-end heatmap prediction instead of gradient-map-based heatmaps, as well as improved heatmap conditioning mechanisms for generators. Dynamically adjusting token budgets during reconstruction and generation is valuable for users with varying computational resources, especially in resource-constrained scenarios involving computation-intensive models such as large-scale image and video generators. Further exploration in this direction would be meaningful.

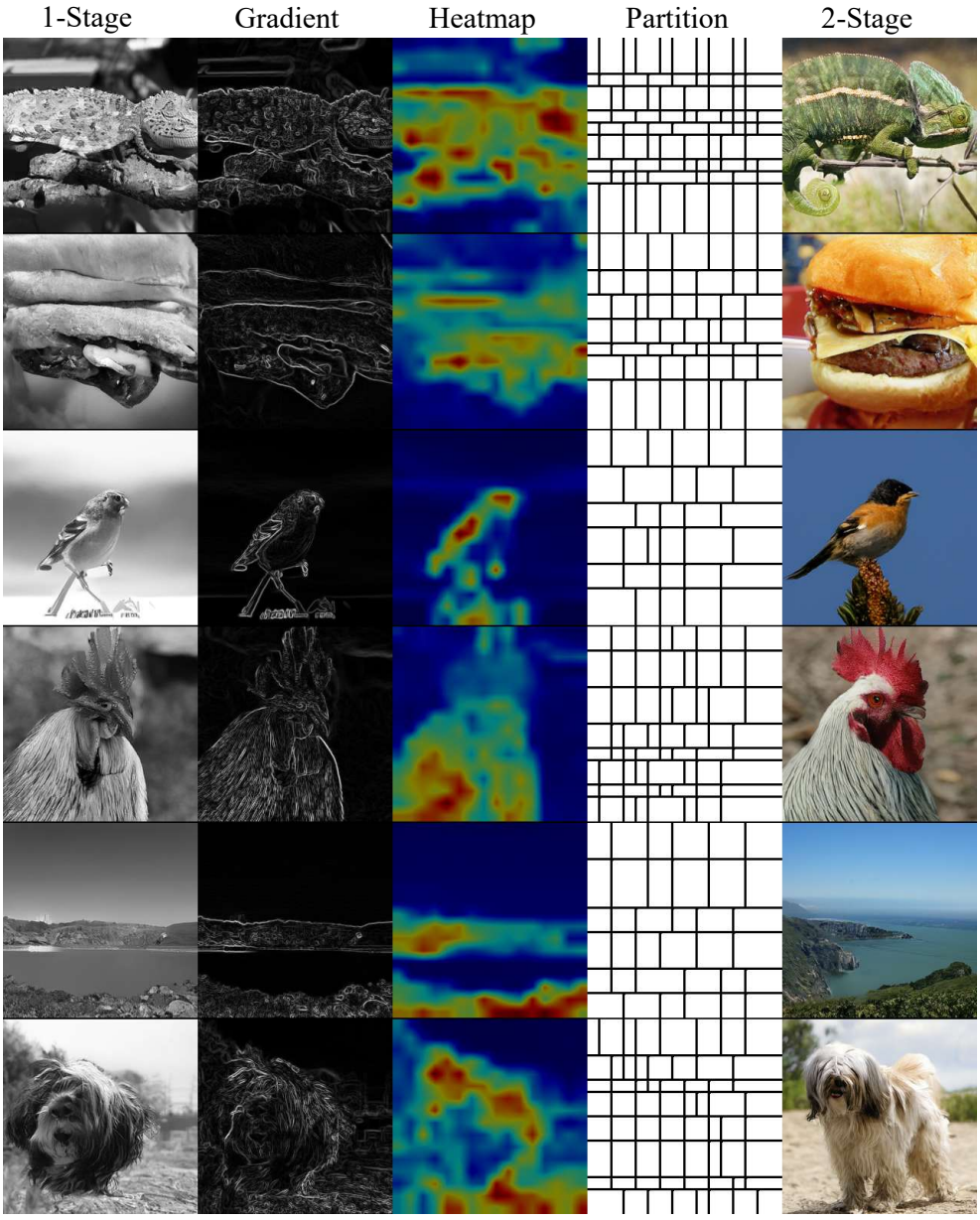


Figure 9: Visualization of generated gray-scale images and corresponding color images.

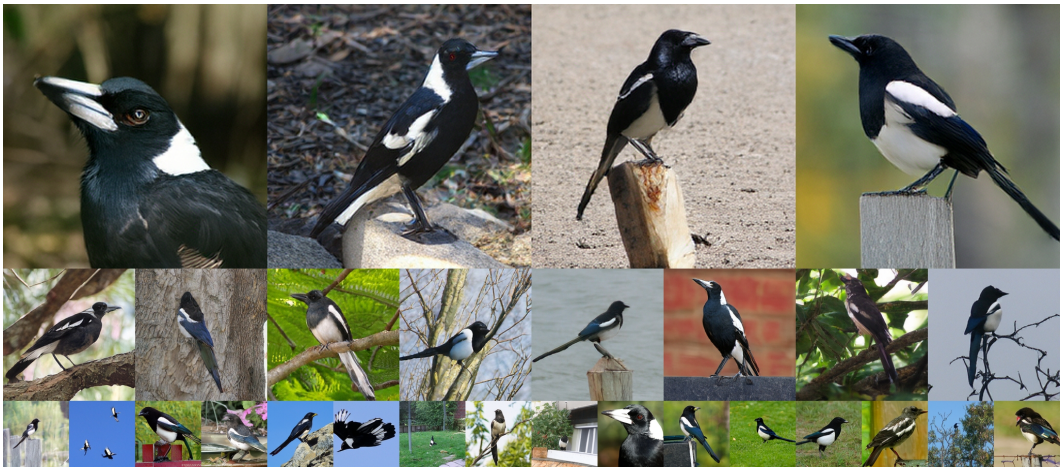


Figure 10: Uncurated 256x256 generation results of DynamicTok + SiT-XL. We use CFG of 3.25. Class label="magpie" (18)



Figure 11: Uncurated 256x256 generation results of DynamicTok + SiT-XL. We use CFG of 3.25. Class label="common iguana, iguana, Iguana iguana" (39)



Figure 12: Uncurated 256x256 generation results of DynamicTok + SiT-XL. We use CFG of 3.25. Class label="lorikeet" (90)



Figure 13: Uncurated 256x256 generation results of DynamicTok + SiT-XL. We use CFG of 3.25. Class label="golden retriever" (207)



Figure 14: Uncurated 256x256 generation results of DynamicTok + SiT-XL. We use CFG of 3.25. Class label="brown bear, bruin, Ursus arctos" (294)



Figure 15: Uncurated 256x256 generation results of DynamicTok + SiT-XL. We use CFG of 3.25. Class label="otter" (360)



Figure 16: Uncurated 256x256 generation results of DynamicTok + SiT-XL. We use CFG of 3.25. Class label="barn" (425)



Figure 17: Uncurated 256x256 generation results of DynamicTok + SiT-XL. We use CFG of 3.25. Class label="passenger car, coach, carriage" (705)



Figure 18: Uncurated 256x256 generation results of DynamicTok + SiT-XL. We use CFG of 3.25. Class label="schooner" (780)



Figure 19: Uncurated 256x256 generation results of DynamicTok + SiT-XL. We use CFG of 3.25. Class label="cliff, drop, drop-off" (972)