SEMANTICALLY STEERABLE IMAGE TOKENIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Adaptive image tokenizers compress an image into a variable number of tokens matched to its complexity, and recent work makes this allocation controllable: conditioned on a target reconstruction error, the tokenizer itself decides how many tokens an image needs. However, the fidelity measure guiding this allocation shapes which aspects of the image are preserved under compression. The pixel-wise error used in prior work is blind to the spatial structure that carries an image's meaning, and under heavy compression such tokenizers are prone to abruptly discarding tokens that encode essential content. We introduce GIST (**G**uided **I**mage **S**emantic **T**okenizer), an adaptive tokenizer that can be steered toward a requested *semantic* fidelity. Its token allocation is jointly controlled by pixel-level and semantic target errors, with semantic error measured in the feature space of a frozen DINOv3 encoder. This joint control enables *semantic protection*: with the semantic target fixed at zero, the reconstruction target can be relaxed to discard pixel detail while preserving what the image depicts. In a controlled comparison on ImageNet-100, GIST outperforms its pixel-controlled counterpart KARL at every token count, with up to 30% lower LPIPS and 37% lower DreamSim. On average, lowering its semantic target reduces LPIPS $2\times$ faster per token than lowering the reconstruction target, and up to $5\times$ in the low-token regime. Semantic fidelity is thus an effective control signal for adaptive tokenization: it lets a tokenizer see, and so preserve, what an image depicts.

> "Humans can recognize the gist of a novel image in a single glance, independent of its complexity."
> — *Oliva & Torralba (2006)*

1 INTRODUCTION

The visual world is inherently heterogeneous: some scenes depict a single object against a plain background, while others capture complex environments dense with detail. Yet the image tokenizers underpinning modern generative models (Esser et al., 2021; Rombach et al., 2022; Chang et al., 2022) are blind to this difference — they allocate the same number of tokens to every image, regardless of its content. This allocation is wasteful: tokenizers exist to compress images into compact latent sequences over which generative modeling becomes tractable, and the length of this sequence directly governs training and inference cost.

A growing line of work therefore makes the token count adaptive (Duggal et al., 2025b; Yan et al., 2025; Miwa et al., 2025; Bachmann et al., 2025). These tokenizers can encode an image into a variable number of tokens, but leave the choice of token count to a trial-and-error search at inference. Ideally, the user would instead specify how faithful the encoding should be and let the tokenizer itself determine the token count. KARL by Duggal et al. (2025a) realizes exactly this: conditioned on the image and a target reconstruction error, it predicts the required token count in a single forward pass and halts the remaining tokens. However, what the tokenizer prioritizes preserving depends on how fidelity is measured. KARL measures fidelity with a pixel-wise reconstruction error that treats pixels independently and is blind to the spatial structure and co-dependence that carry an image's meaning. An encoding can therefore meet the pixel target yet fail to preserve what the image depicts. For example, a small bird against an open sky can become an unrecognizable blob of similar colors

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.

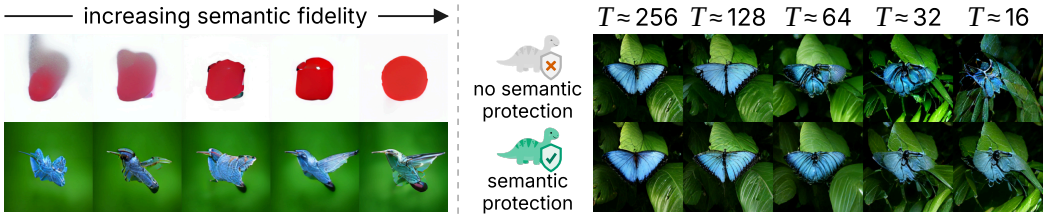


Figure 1: **Semantic control unlocks semantic protection.** (*Left*) GIST can be steered toward a requested semantic fidelity in DINOv3 feature space, as illustrated here with a red circle (top) and a hummingbird (bottom). Raising the requested fidelity steers the encoding from little more than the image's color distribution to a reconstruction that preserves what the image depicts. (*Right*) Requesting full semantic fidelity turns this control into a guardrail: without it, compression into fewer tokens disintegrates the butterfly; with it, the subject remains recognizable down to 16 tokens.

yet contribute little to the overall pixel error. Indeed, we experimentally observe that under heavy token compression, KARL is prone to abruptly halting tokens that encode essential image content.

We introduce GIST (**G**uided **I**mage **S**emantic **T**okenizer), an adaptive image tokenizer that can be steered toward a requested *semantic* fidelity. Its token allocation is jointly controlled by a reconstruction target error ϵ_{rec} and a semantic target error ϵ_{sem}, with semantic error measured in the feature space of a frozen DINOv3 encoder (Siméoni et al., 2026). These targets allow pixel-level and semantic fidelity to be specified independently: lowering ϵ_{rec} demands tokens that reproduce the image's exact pixels, while lowering ϵ_{sem} demands tokens that preserve what the image depicts.

As illustrated in Figure 1, this two-dimensional control unlocks a new capability we call *semantic protection*: fixing the semantic target error at zero turns it into a guardrail under which the reconstruction target can be relaxed far more aggressively. The encoder then discards tokens encoding redundant pixel detail while retaining those needed to preserve the image's semantic content, as captured in DINOv3 feature space.

We validate GIST on ImageNet-100 against KARL, with all models trained from scratch under identical conditions. GIST Pareto-improves reconstruction quality at every number of active tokens, with up to 30% lower LPIPS and 37% lower DreamSim. We further find that lowering the semantic target is more token-efficient than lowering the reconstruction target. Per token, it reduces LPIPS $2\times$ faster on average, and up to $5\times$ in the low-token regime.

We find that these gains do not come from DINOv3 conditioning alone. A variant with just the semantic target merely matches the baseline on reconstruction, yet its tokens are the most semantically informative of all models we train. This suggests a division of labor. Conditioning on DINOv3 feature error structures the latent space so that token complexity reflects the *meaning* of an image rather than only its pixel-level appearance. Semantic protection converts this structure into reconstruction gains. The takeaway is simple: an adaptive tokenizer can only preserve what its fidelity measure can see, and a semantic measure lets it see what the image depicts.

2 Related Work

Image Tokenization. Image tokenizers compress images into compact latent representations over which generative models can be efficiently trained. Modeling a distribution directly in pixel space is difficult due to its high dimensionality, much of which encodes perceptually irrelevant detail. Tokenizers address this by mapping images to a lower-dimensional latent space that retains only the information necessary for faithful reconstruction. Variational Autoencoders (VAEs) (Kingma & Welling, 2014) learn such representations as continuous latents and form the basis of latent diffusion models (Rombach et al., 2022). VQ-VAE (van den Oord et al., 2017; Razavi et al., 2019) and VQGAN (Esser et al., 2021) instead apply vector quantization to produce discrete-valued tokens and have become the de facto tokenizers for masked (Chang et al., 2022) and autoregressive (Esser et al., 2021; Tian et al., 2024; Sun et al., 2024) image generation. All of these tokenizers, however, share one structural constraint: the latent is a 2D grid whose size is fixed by the image resolution.

1D and Adaptive Tokenization. TiTok (Yu et al., 2024) departs from the 2D grid: a ViT encoder distills the image into a short 1D sequence of register tokens (Darcet et al., 2024). The sequence length, however, is still fixed at training time, independent of image content. Subsequent work makes the length adaptive: ALIT (Duggal et al., 2025b) recurrently grows the 1D sequence, while ElasticTok (Yan et al., 2025), One-D-Piece (Miwa et al., 2025), FlexTok (Bachmann et al., 2025), and SMAP (Li et al., 2026) train with tail-token dropout (Rippel et al., 2014) so that any prefix of the ordered, coarse-to-fine token sequence remains decodable. These methods make variable-length representations possible, but leave the choice of length to a search at inference: the user or the method itself must encode the image at several budgets and keep the shortest that suffices.

CAT (Shen et al., 2025) sidesteps this search by scoring image complexity with an LLM and routing each image to one of three grid resolutions, yet the budget is set by an external predictor, at coarse granularity, and cannot be conditioned on a requested fidelity. KARL (Duggal et al., 2025a) instead makes the tokenizer itself decide: conditioned on the image and a target reconstruction quality, it predicts the required token count in a single forward pass and halts the remaining tokens. GIST inherits KARL's single-pass, quality-conditioned design but extends the notion of quality itself: where KARL targets pixel-space reconstruction error alone, GIST adds a second, semantic target, which we find improves token efficiency in the low-token regime.

Semantic Alignment of Tokenizers. A separate line of work distills the semantic knowledge of pretrained vision encoders into tokenizers. An influential precursor is REPA (Yu et al., 2025), which aligns intermediate features of a diffusion transformer with DINOv2 (Oquab et al., 2024) features and accelerates convergence by more than an order of magnitude. VA-VAE (Yao et al., 2025) moves the alignment from the generator into the tokenizer: a foundation-model alignment loss structures the VAE's latent space during training, which preserves reconstruction quality while making the latents far easier for diffusion transformers to model (see also Qu et al., 2026). VFMTok (Zheng et al., 2025) and RAE (Zheng et al., 2026) take this idea further by building the tokenizer directly on frozen foundation model features. Adaptive-length tokenizers have adopted this alignment only in the decoder: FlexTok and Semanticist (Wen et al., 2025) train their diffusion decoders with a REPA loss, and ReTok (Fu et al., 2026) decays a decoder-side alignment loss along its ordered token sequence.

In all of these methods, however, semantic alignment is a fixed training-time bias: it shapes what the model encodes and improves output quality at a given token count, but plays no role in deciding how many tokens an image receives. GIST instead makes semantics an explicit control signal: the tokenizer is conditioned on a target quality in DINOv3 (Siméoni et al., 2026) feature space, so the token count is steered by the requested semantic fidelity rather than by pixel fidelity alone. DINOv3 thus serves purely to *measure* semantic fidelity for this conditioning — its features never supervise the tokens themselves, unlike the alignment approaches above.

3 METHOD

GIST builds on KARL (Duggal et al., 2025a), a single-pass adaptive 1D tokenizer that allocates a variable number of tokens per image based on a single reconstruction target error. We introduce a second conditioning signal: a semantic target error, measured in the feature space of a frozen DINOv3 encoder (Siméoni et al., 2026). This expands KARL's one-dimensional fidelity control into a two-dimensional space in which pixel-level and semantic fidelity can be targeted independently. We illustrate the idea in Figure 2. We describe the architecture and the training procedure below, marking where GIST deviates from KARL (in this color).

3.1 GIST ARCHITECTURE

GIST follows a standard encoder-decoder architecture that maps images to a 1D, variable-length latent space. Given an input image, we first extract 2D patch embeddings $\boldsymbol{X}_{\text{2D}} \in \mathbb{R}^{N \times D}$ from a pretrained VQGAN. The encoder then distills these embeddings into learnable register tokens $\boldsymbol{Z}_{\text{1D}} \in \mathbb{R}^{T_{\max} \times D}$ (Darcet et al., 2024), following the 2D-to-1D token distillation of TiTok (Yu et al., 2024) adopted by KARL. The register tokens act as read-write storage during self-attention.

To control the complexity of the 1D tokenization, KARL conditions the encoder on a reconstruction **target error** ϵ_{rec}, governing pixel-level fidelity. We introduce a second, semantic target error ϵ_{sem},

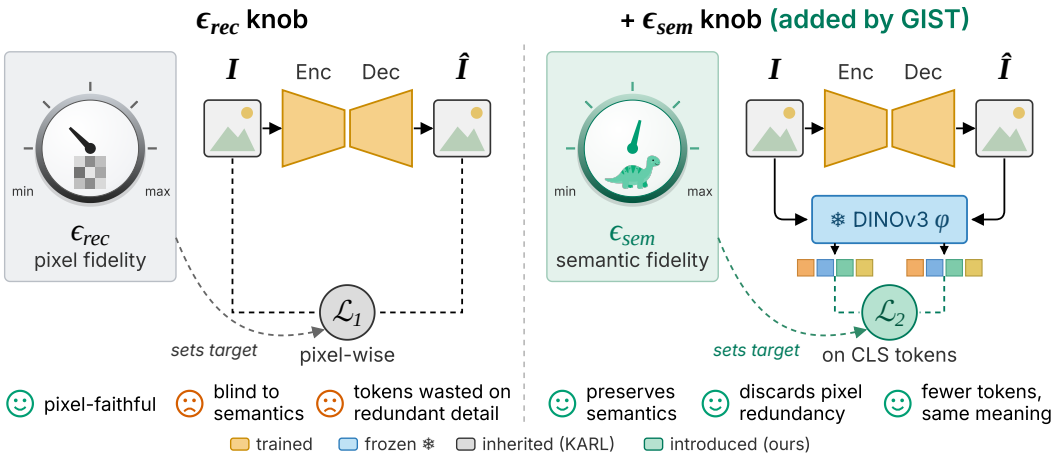


Figure 2: **GIST makes semantic fidelity an explicit control signal.** (*Left*) KARL (Duggal et al., 2025a) steers the tokenizer with a single reconstruction target ϵ_{rec} for the pixel-wise L_1 error between input $\boldsymbol{I}$ and its reconstruction $\hat{\boldsymbol{I}}$: encodings become pixel-faithful, but the error is blind to semantics, so tokens are prone to be wasted on redundant detail. (*Right*) GIST keeps ϵ_{rec} and adds a second, semantic target ϵ_{sem} for the error between DINOv3 CLS features of $\boldsymbol{I}$ and $\hat{\boldsymbol{I}}$: lowering ϵ_{sem} demands tokens that preserve what the image depicts, so the encoder can discard redundant pixel detail.

governing perceptual fidelity in the feature space of a frozen DINOv3 encoder (Siméoni et al., 2026). Each target specifies an upper bound on its corresponding error: lowering it demands a more faithful encoding, and in turn more tokens. Importantly, the two targets capture faithfulness of different kinds: lowering ϵ_{rec} steers the encoder toward tokens that reproduce the image's exact pixel values, while lowering ϵ_{sem} steers it toward tokens that preserve its semantic identity (what the image depicts, rather than the exact pixels used to depict it). The targets thus control not only *how many* tokens an image requires, but also *what* those tokens encode.

We convert these continuous target errors into learnable conditioning signals by mapping them to the indices of their nearest valid upper-bound bins and embedding them via learnable dictionaries $\boldsymbol{E}_{\text{rec}}$ and $\boldsymbol{E}_{\text{sem}}$ to produce *target embeddings*:

$$\boldsymbol{\epsilon}_{\text{rec}} = \boldsymbol{E}_{\text{rec}}[\text{bin}(\epsilon_{\text{rec}})], \tag{1}$$

$$\boldsymbol{\epsilon}_{\text{sem}} = \boldsymbol{E}_{\text{sem}}[\text{bin}(\epsilon_{\text{sem}})]. \tag{2}$$

The encoder attends over the concatenated target embeddings, patch embeddings, and register tokens, allowing the registers to absorb information from all other tokens via self-attention. We retain only the updated register outputs $\boldsymbol{Z}'_{\text{1D}}$:

$$\boldsymbol{Z}'_{\text{1D}} = \text{Enc}\left([\boldsymbol{\epsilon}_{\text{rec}}; \boldsymbol{\epsilon}_{\text{sem}}; \boldsymbol{X}_{\text{2D}}; \boldsymbol{Z}_{\text{1D}}]\right)_{\boldsymbol{Z}_{\text{1D}}}. \tag{3}$$

Depending on image complexity, only a subset of register tokens may be required to meet the target error bounds $(\epsilon_{\text{rec}}, \epsilon_{\text{sem}})$. To select the most suitable subset, a halting MLP (Duggal et al., 2025a) predicts a per-token halting probability from $\boldsymbol{Z}'_{\text{1D}}$. Tokens whose halting probability exceeds a predefined threshold (e.g., $\tau = 0.75$) are marked inactive and excluded from decoding.

$$\boldsymbol{\omega} = \sigma\left(\text{MLP}_{\text{halt}}(\boldsymbol{Z}'_{\text{1D}})\right) \in [0,1]^{T_{\max}}, \tag{4}$$

$$\boldsymbol{m}_{\text{halt}} = \mathbb{I}(\boldsymbol{\omega} > \tau) \in \{0,1\}^{T_{\max}}. \tag{5}$$

To prevent halted latents from interacting with the active sequence, we derive a joint self-attention mask $\boldsymbol{M}$ from $\boldsymbol{m}_{\text{halt}}$ that blocks attention to and from any halted token. Under this mask, the decoder reconstructs the VQGAN patch embeddings by jointly attending over the quantized latent tokens $\hat{\boldsymbol{Z}}_{\text{1D}} = \text{Quant}(\boldsymbol{Z}'_{\text{1D}})$ and a 2D grid of learnable query tokens $\tilde{\boldsymbol{X}}_{\text{2D}}$:

$$\hat{\boldsymbol{X}}_{\text{2D}} = \text{Dec}\left([\tilde{\boldsymbol{X}}_{\text{2D}}; \hat{\boldsymbol{Z}}_{\text{1D}}], \text{mask} = \boldsymbol{M}\right)_{\tilde{\boldsymbol{X}}_{\text{2D}}}. \tag{6}$$

At inference, tokenization requires only a single forward pass: the user specifies the target errors $(\epsilon_{\text{rec}}, \epsilon_{\text{sem}})$, and the encoder produces and halts tokens accordingly. This requires neither iterative re-encoding nor a search over token counts.

3.2 Dual Forward Pass Training

The architecture of GIST allows us to steer the complexity of the image encoding by conditioning on reconstruction and semantic target errors. The values of these targets influence *how* the encoder constructs suitable tokens $\boldsymbol{Z}'_{1\mathrm{D}}$ to meet the target error requirements, as well as *how* the halting MLP processes these tokens to decide whether each of them should be halted or not. This allows our method to dynamically infer both the appropriate number of tokens and the nature of each token for any given image, without resorting to trial-and-error or manual tuning. *How, then, can we train our method to be directly controllable by these targets?*

We achieve this by following Duggal et al. (2025a) and optimizing GIST using a dual forward pass based on the principles of upside-down reinforcement learning (Schmidhuber, 2019). During the **first forward pass**, we set ϵ_{rec} and ϵ_{sem} to 0 to let GIST aim for a perfect reconstruction under a uniformly sampled token budget $T \sim U(T_{\min}, T_{\max})$. We measure the error GIST actually realizes in *pixel space* by passing $\hat{\boldsymbol{X}}_{2\mathrm{D}}$ through the VQGAN decoder. Whereas KARL measures the realized error solely through a pixel-wise reconstruction error, we additionally introduce a semantic error $\epsilon^*_{\mathrm{sem}}$ that captures perceptual fidelity in the feature space of a frozen DINOv3 encoder:

$$\epsilon^*_{\mathrm{rec}} = \tfrac{1}{3HW} \left\| \boldsymbol{I} - \mathrm{VQGAN}_{\mathrm{dec}}(\hat{\boldsymbol{X}}_{2\mathrm{D}}) \right\|_1, \tag{7}$$

$$\epsilon^*_{\mathrm{sem}} = \tfrac{1}{D_\phi} \left\| \phi(\boldsymbol{I}) - \phi\big(\mathrm{VQGAN}_{\mathrm{dec}}(\hat{\boldsymbol{X}}_{2\mathrm{D}})\big) \right\|_2^2, \tag{8}$$

where $\boldsymbol{I} \in [0,1]^{3\times H\times W}$ is the input image and $\phi : \mathbb{R}^{3\times H\times W} \to \mathbb{R}^{D_\phi}$ extracts the post-LayerNorm CLS token from the frozen DINOv3-L/16 image encoder. These realized target errors $\epsilon^* = (\epsilon^*_{\mathrm{rec}}, \epsilon^*_{\mathrm{sem}})$ serve as a task-specifying input reward for the second forward pass. Intuitively, satisfying error bound ϵ^* requires no more than T tokens for the given image.

To leverage this insight, the **second forward pass** conditions the encoder on ϵ^* and expands the token budget by ΔT tokens, which the model is trained to disregard. Formally, the halting loss (Duggal et al., 2025a) is defined as:

$$\mathcal{L}_{\mathrm{halt}}(\boldsymbol{\omega}) = \mathrm{BCE}(\boldsymbol{\omega}_{0:T}, \mathbf{0}) + \mathrm{BCE}(\boldsymbol{\omega}_{T:T+\Delta T}, \mathbf{1}). \tag{9}$$

This loss encourages the model to assign low halting probabilities to essential tokens ($\boldsymbol{\omega}_{0:T} \to \mathbf{0}$), and high halting probabilities to the redundant ones ($\boldsymbol{\omega}_{T:T+\Delta T} \to \mathbf{1}$). By doing so, the model implicitly learns the relationship between the input image, the specified error bound ϵ^*, and the requisite number of tokens T.

During both passes, we optimize GIST using a single objective:

$$\mathcal{L} = \mathcal{L}_{\mathrm{rec}}(\hat{\boldsymbol{X}}_{2\mathrm{D}}, \boldsymbol{X}_{2\mathrm{D}}) + \beta\, \mathcal{L}_{\mathrm{quant}}(\hat{\boldsymbol{Z}}_{1\mathrm{D}}) + \lambda\, \mathcal{L}_{\mathrm{halt}}(\boldsymbol{\omega}), \tag{10}$$

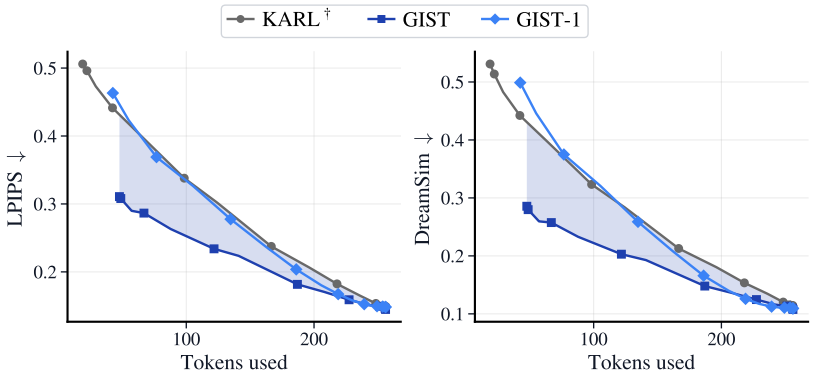
where $\mathcal{L}_{\mathrm{rec}}(\hat{\boldsymbol{X}}_{2\mathrm{D}}, \boldsymbol{X}_{2\mathrm{D}})$ is the reconstruction loss against the VQGAN tokenizer, $\mathcal{L}_{\mathrm{quant}}(\hat{\boldsymbol{Z}}_{1\mathrm{D}})$ is the standard VQ-VAE (van den Oord et al., 2017) commitment and codebook loss on the register outputs $\boldsymbol{Z}'_{1\mathrm{D}}$, and $\mathcal{L}_{\mathrm{halt}}(\boldsymbol{\omega})$ is the halting loss defined above. We set $\beta = \lambda = 1$ throughout this paper.

Note: The two passes share the main loss objective by construction: in the first pass we have $\Delta T = 0$, so the slice $\boldsymbol{\omega}_{T:T+\Delta T}$ is empty and $\mathcal{L}_{\mathrm{halt}}$ reduces to $\mathrm{BCE}(\boldsymbol{\omega}_{0:T}, \mathbf{0})$ to push all T halting probabilities toward zero. In the second pass, both terms of $\mathcal{L}_{\mathrm{halt}}$ are active to teach the model to halt the ΔT redundant tokens while keeping the first T tokens active.

Training Details. We train GIST in two sequential stages: Latent Distillation Pre-Training and Pixel-Space Finetuning, both on ImageNet-100 (further details in Appendix A.1). Full training requires approximately 470 H100 GPU-hours on 8 NVIDIA H100 GPUs (≈2.5 days wall-clock).

4 Experiments

In this section, we explore how modeling an image tokenizer's complexity along a semantic axis affects tokenizer behavior and performance. We evaluate GIST against KARL in terms of reconstruction quality, and visually compare how the reconstruction and semantic targets affect the image reconstructions for both methods. Subsequently, we study the role of the semantic control in GIST.



ϵ_{rec}	0.05	0.10	0.16
Tokens used ↓			
KARL†	**236**	**124**	**42**
GIST	242	141	67
LPIPS ↓			
KARL†	0.16	0.30	0.44
GIST	**0.15**	**0.22**	**0.29**
DreamSim ↓			
KARL†	0.13	0.28	0.44
GIST	**0.12**	**0.19**	**0.26**

Figure 3: **Adding semantic control Pareto-improves reconstruction quality at every token count.** (*Left*) Pareto fronts on the ImageNet-100 validation set, obtained by varying ϵ_{rec} for KARL† and GIST (fixing $\epsilon_{\text{sem}} = 0$), and by varying ϵ_{sem} for GIST-1. The shaded regions highlight GIST's reduction in reconstruction error relative to KARL† at matched token counts. (*Right*) Tokens used, LPIPS, and DreamSim for KARL† and GIST at equal reconstruction targets, with the best values in bold. We report the same comparison on the full set of reconstruction quality metrics in Appendix D.

4.1 Setup

Model Variants. We consider two variants of our method: (i) GIST, which is conditioned on both a reconstruction and a semantic target error, and (ii) GIST-1, which is conditioned *only* on a semantic target error. GIST decomposes complexity into two complementary axes and lets us control each independently, whereas GIST-1 replaces the reconstruction target with the semantic one and lets us isolate their individual behavior.

Baselines. We evaluate GIST against KARL as our primary baseline. We note a slight performance discrepancy between the authors' published checkpoint and the model produced by their released training code. To ensure a controlled comparison that isolates the effect of our contributions, all evaluated models are trained from scratch under identical conditions using the official, unmodified KARL training script. We denote the resulting baseline as KARL†. We document all reproduction details in Appendix B, and situate GIST among other tokenizers in Appendix C. Code, training configurations, and all trained checkpoints will be released upon publication.

Data and Evaluation. All models are trained on ImageNet-100 (Tian et al., 2020) and, unless stated otherwise, evaluated on its validation set. Following KARL, we focus on per-image metrics. Because the targets set per-image error bounds, the fidelity of each reconstruction to its own input is the quantity of interest. Distribution-level metrics such as rFID (Heusel et al., 2017) instead score whether reconstructions are realistic *as a set*, and can reward a plausible image that is not a faithful reconstruction (see Appendix C). We provide training and evaluation details in Appendix A.

4.2 The Value of Semantic Steerability

GIST Pareto-improves reconstruction quality. In Figure 3, we compare GIST, GIST-1, and KARL† along the token-fidelity Pareto front in LPIPS (Zhang et al., 2018) and DreamSim (Fu et al., 2023), obtained by varying each method's target values. We draw two conclusions. First, GIST achieves better reconstruction quality than KARL† at every token count. The gap widens as images are compressed into fewer tokens, with up to 30% lower LPIPS and 37% lower DreamSim. This reconstruction advantage also extends to the multi-object scenes of MS COCO (Lin et al., 2014), without further training or adaptation (Appendix E). Second, GIST-1, which is conditioned only on the semantic target, performs on par with KARL†. This suggests that DINOv3 conditioning improves reconstruction quality only when complexity is disentangled into a reconstructive and a semantic axis. We attribute the gap to the semantic target acting as a guardrail: fixing $\epsilon_{\text{sem}} = 0$ while relaxing ϵ_{rec} lets the encoder discard pixel-level detail while preserving the image's semantic identity. We verify this in the remainder of our experiments.

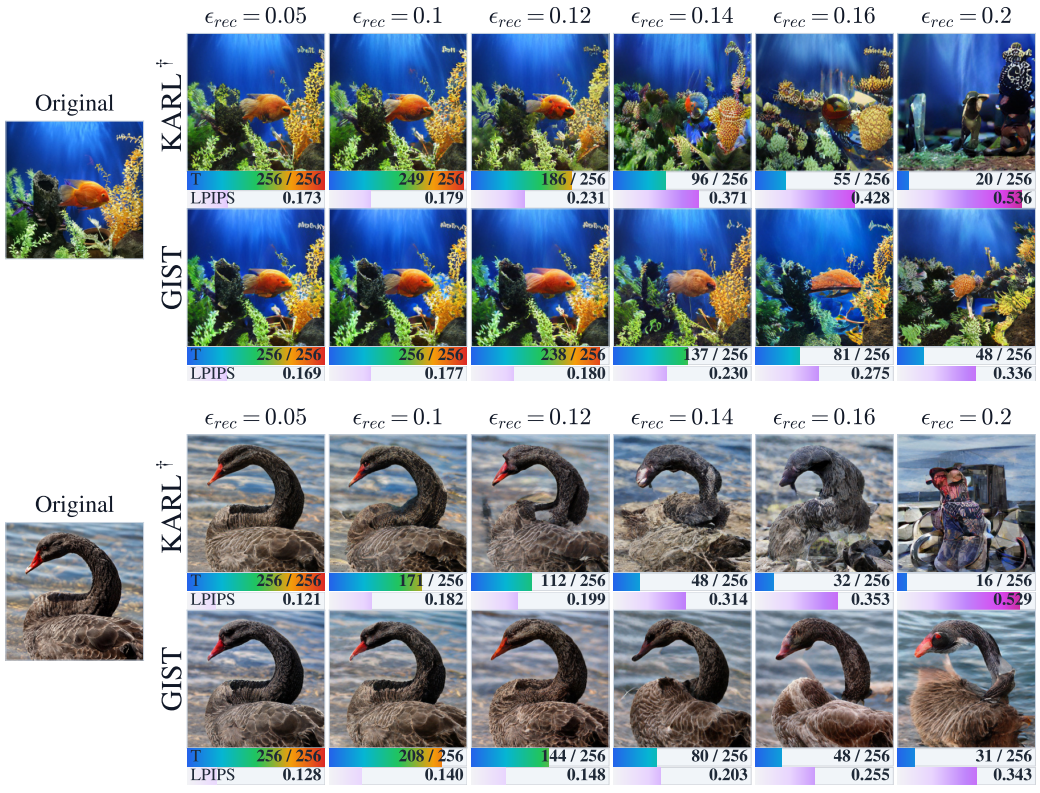


Figure 4: **Semantic protection keeps the subject recognizable under heavy token compression.** For each sample (*Goldfish reef*, top; *Black swan*, bottom), the upper row shows KARL† reconstructions as we vary ϵ_{rec}, and the lower row shows GIST reconstructions with ϵ_{sem} fixed at 0 and ϵ_{rec} varied over the same values. The far-left image is the original; token budget is fixed at $T = 256$; per-cell strips report the active token count and LPIPS. As more tokens are halted, KARL† disintegrates the subject into artifacts that retain little beyond its color distribution, while GIST keeps it recognizable. See Appendix G for additional samples.

In Appendix F, we repeat this comparison while varying the input token budget instead of the error targets. We find that GIST, GIST-1, and KARL† perform on par in this setting, suggesting that the main benefit of our method comes from how conditioning on these targets organizes the latent space along complementary axes of complexity, rather than from base tokenizer capacity.

GIST unlocks semantic protection. To test the guardrail hypothesis visually, we qualitatively compare KARL† and GIST in Figure 4. As we raise the reconstruction target, we observe that KARL† is prone to abruptly halting tokens that encode essential image content. For *Goldfish reef*, the fish becomes unrecognizable as soon as a substantial number of tokens are halted (here, $\epsilon_{\text{rec}} = 0.14$), and the image progressively collapses into incoherent artifacts that retain little of the original image beyond its color distribution. We observe the same failure for *Black swan*, where KARL† preserves only the water background while the swan disintegrates into jigsaw-like fragments that retain only its color distribution. GIST, by contrast, is capable of *reliably* halting more tokens before incurring such artifacts. With $\epsilon_{\text{sem}} = 0$, raising ϵ_{rec} lets the encoder discard tokens encoding redundant pixel details while *protecting* those indispensable to the image's semantic identity. Across both samples and those in Appendix G, this behavior is consistent — the image's core gist survives more reliably even under heavy token compression.

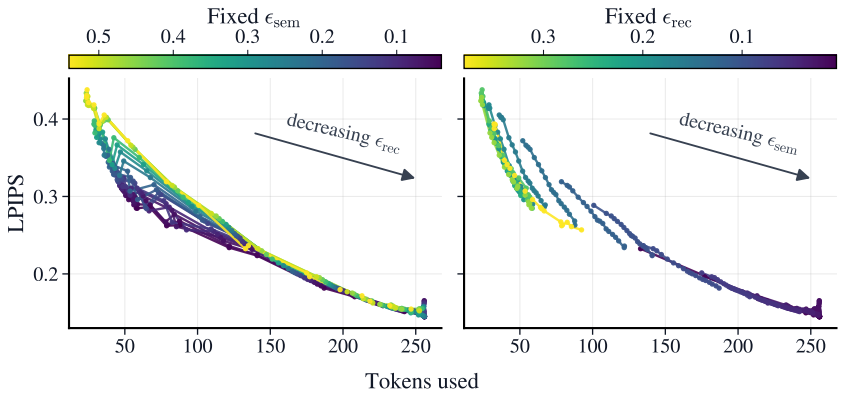


Figure 5: **Semantic control reduces LPIPS $2\times$ faster per token than pixel-level control.** LPIPS trajectories of GIST as we vary one target with the other held fixed (shown by color). Lowering ϵ_{rec} (left) traces long, shallow curves that span nearly the full token range, whereas lowering ϵ_{sem} (right) drops LPIPS sharply over only a few added tokens.

4.3 Understanding the Role of Semantic Control

Semantic control is more token-efficient. To better understand the role of each target in GIST, we isolate their individual effects by holding one target (ϵ_{rec} or ϵ_{sem}) fixed while gradually decreasing the other to trace the resulting LPIPS-token trajectories (Figure 5). Varying the two targets produces qualitatively different trajectories: lowering ϵ_{rec} (left) yields long, shallow curves that span nearly the full token range, whereas lowering ϵ_{sem} (right) yields steep, segmented curves that drop LPIPS sharply over only a few added tokens. In other words, lowering the semantic target is more token-efficient at reducing LPIPS than lowering the reconstruction target. On average, it reduces LPIPS $2\times$ faster per token, and up to $5\times$ faster in the low-token regime.

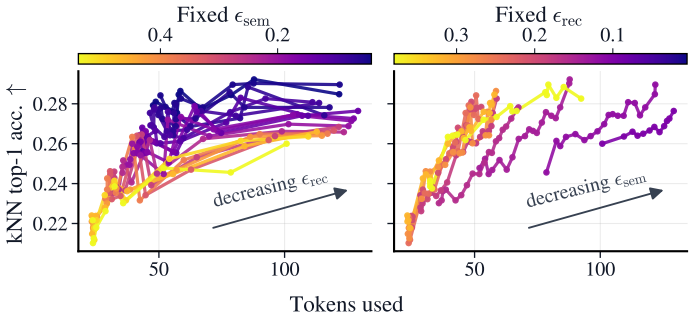
This efficiency is consistent with the semantic protection we observed in Figure 4: we hypothesize that semantic control is more token-efficient because it prioritizes the tokens that carry the image's core gist. The reconstruction target, by contrast, specifies a bound on pixel-wise L_1 error, which is blind to co-dependence and spatial structure that carry meaning in a visual signal. Because this error provides no meaningful basis for prioritization, the encoder allocates tokens to faithfully reconstruct details that contribute little to perceptual quality.

Semantic conditioning makes tokens more semantically informative. Thus far, we evaluated GIST through the quality of its reconstructions. We now turn our attention to the latent tokens themselves and ask how conditioning on ϵ_{sem} affects their semantic informativeness. In Figure 6a, we compare the kNN classification accuracy of GIST, GIST-1, and KARL† as we vary each method's target values (protocol in Appendix A.3). GIST and GIST-1 both achieve a higher kNN accuracy than KARL† at any number of active tokens (+3–7 and +7–17 points, respectively, across 40–256 tokens). Notably, GIST-1 performs best of the three, despite performing on par with KARL† and worse than GIST on reconstruction quality (Figure 3). The same ordering holds under a trained attentive probe, with narrower but consistent gaps between methods (Appendix K). We also visualize GIST's kNN-token trajectories as we vary ϵ_{rec} and ϵ_{sem} in isolation to study their individual effect in GIST's latent space in Figure 6b. Consistent with our earlier findings, lowering ϵ_{sem} increases kNN accuracy at a much steeper rate per added token than lowering ϵ_{rec}.

Together, these results suggest that conditioning on ϵ_{sem} structures the latent space of GIST and GIST-1 so that token complexity reflects the *meaning* of an image rather than only its pixel-level appearance. However, this semantic structure can only be used to improve reconstruction quality when we introduce ϵ_{sem} as a second target, where it can act as a guardrail that protects the image's semantic identity while relaxing ϵ_{rec} allows the encoder to discard redundant pixel detail (Figures 3 and 4). In other words, semantic protection converts this structure into reconstruction gains.

Method	Tokens 64	128	192
kNN top-1 acc. (%) ↑			
KARL†	21.9	23.3	24.2
GIST-1	**33.9**	**37.0**	**39.0**
GIST	28.0	28.4	28.6

(a) GIST vs. KARL† vs. GIST-1



(b) Varying one target at a time

Figure 6: **Semantic conditioning makes tokens more semantically informative.** (a) kNN accuracy of GIST, KARL†, and GIST-1 as we vary each method's target values (same setup as Figure 3), with accuracies linearly interpolated at matched mean active token counts. GIST and GIST-1 outperform KARL† at every token count, and GIST-1 performs best despite performing on par with KARL† and worse than GIST on reconstruction quality. (b) kNN accuracy trajectories of GIST as we vary one target with the other held fixed (shown by color). Lowering ϵ_{sem} increases kNN accuracy at a much steeper rate than lowering ϵ_{rec}. Accuracies are computed by mean pooling over the active tokens; Appendix I reports the same comparison under max pooling.

5 Conclusion

We have presented GIST, an adaptive image tokenizer that can be steered toward a requested *semantic* fidelity. Its token allocation is jointly controlled by a pixel-wise reconstruction target and a semantic target in DINOv3 feature space, allowing pixel-level and semantic fidelity to be specified independently. Our experiments point to a division of labor between the two roles the semantic target plays. During training, the DINOv3 target structures the latent space so that token complexity reflects the meaning of an image rather than only its pixel-level appearance. At inference, it acts as a guardrail: with ϵ_{sem} fixed at zero, the reconstruction target can be relaxed freely while the encoder protects the tokens indispensable to the image's semantic identity. Under this guardrail, GIST Pareto-improves reconstruction quality over KARL† at every token count, and its tokens themselves become more semantically informative.

Our work also has its limitations. First, our evaluation is deliberately controlled rather than competitive: all models are trained from scratch on ImageNet-100 over a pretrained VQGAN base. Accordingly, we claim no state of the art. The evidence supports the narrower point that semantic fidelity is an effective control signal for adaptive tokenization. Second, the same principle that motivates GIST also bounds it: a tokenizer can only preserve what its fidelity measure can see, and we instantiate that measure with a single global feature, the DINOv3 CLS token. For images whose meaning is distributed across many regions, such as a document page or a busy scene in which several objects matter equally, one global summary may under-specify what to protect. Patch-level semantic targets are a natural extension.

More broadly, we see the DINOv3 target as one instance of a general recipe: any measure of what matters can serve as a conditioning target. Because the measure only conditions the tokenizer and is never differentiated through, it need not even be differentiable. A different vision encoder, a text encoder, or a task-specific score such as detection or OCR accuracy could take its place. In this way, GIST can be tailored to the kind of information its tokens should preserve. The tokens themselves also point beyond reconstruction. Semantic alignment is known to ease generative training (Yu et al., 2025; Yao et al., 2025), and we suspect GIST's tokens offer a favorable interface for generative models trained on top of them. Whether semantic steerability carries over to generation is a question we leave open for future work.

Humans recognize the gist of a novel image in a single glance, independent of its complexity; we hope GIST is a step toward tokenizers that see images the same way.

AI USE STATEMENT

In this work, we used generative AI tools (Anthropic's Claude, including the Claude Code coding assistant) for the following tasks in the required-disclosure category: implementing methods, in the form of software supporting our experiments (evaluation and job scripts, and experiment execution and management on our compute cluster); interpreting results, in the form of computing the aggregate numerical values reported in the text (e.g., averages over per-setting metrics) programmatically from the experiment output files; and designing or providing feedback on research methodology or experiments, in the form of brainstorming individual design choices (e.g., suggesting isocurves as the visualization of the four-dimensional relation between the two target qualities and the resulting token counts and reconstruction quality, and discussing which attentive-probe design to use), with the final choices made by us.

We have not used generative AI tools to help develop theoretical models or conceptual frameworks, to propose or refine hypotheses, or to generate synthetic datasets; and the rest of the required disclosure tasks (formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, assisting with translation, cleaning and reformatting datasets, and supporting qualitative and thematic data analysis) are not applicable to this work.

Additionally, we used generative AI tools for tasks in the recommended-disclosure category: drafting and iteratively revising parts of the paper to improve clarity and readability; proposing candidate titles (the final title is our own); designing the introduction and method-overview figures, and creating the scripts that generate our plots and result tables; creating and editing software code; suggesting experimental parameters; summarizing or analyzing existing literature, in the form of answering targeted questions about the content of individual papers; identifying relevant literature, in the form of surfacing a few additional related-work references; and reformatting bibliography entries, in the form of removing redundant fields from entries we collected from DBLP (generative AI was not used to generate the citation entries themselves).

We have reviewed all AI-assisted work: we read, revised, and approved every AI-drafted sentence; we verified and tested AI-generated code for correctness; all AI-computed numbers and tables were derived programmatically from the experiment result files (never produced from model memory) and we verified them against those files; and we manually inspected AI edits to the bibliography line by line. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

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

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mido Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *Trans. Mach. Learn. Res.*, 2024.

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*, 2022.

Min Jin Chong and David A. Forsyth. Effectively unbiased FID and inception score and where to find them. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *International Conference on Learning Representations*, 2024.

Shivam Duggal, Sanghyun Byun, William T. Freeman, Antonio Torralba, and Phillip Isola. Single-pass adaptive image tokenization for minimum program search. In *Advances in Neural Information Processing Systems*, 2025a.

Shivam Duggal, Phillip Isola, Antonio Torralba, and William T. Freeman. Adaptive length image tokenization via recurrent allocation. In *International Conference on Learning Representations*, 2025b.

Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. DreamSim: Learning new dimensions of human visual similarity using synthetic data. In *Advances in Neural Information Processing Systems*, 2023.

Zixuan Fu, Lanqing Guo, Chong Wang, Binbin Song, Ding Liu, and Bihan Wen. Improving flexible image tokenizers for autoregressive image generation. *CoRR*, abs/2601.01535, 2026.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, 2017.

Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.

Qingfeng Li, Haoxian Zhang, Xu He, Songlin Tang, Zhixue Fang, Xiaoqiang Liu, Pengfei Wan, and Guoqi Li. Semantic-aware prefix learning for token-efficient image generation. *CoRR*, abs/2603.25249, 2026.

Tianhong Li, Huiwen Chang, Shlok Kumar Mishra, Han Zhang, Dina Katabi, and Dilip Krishnan. MAGE: Masked generative encoder to unify representation learning and image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision*, 2014.

Keita Miwa, Kento Sasaki, Hidehisa Arai, Tsubasa Takahashi, and Yu Yamaguchi. One-D-Piece: Image tokenizer meets quality-controllable compression. *CoRR*, abs/2501.10064, 2025.

Aude Oliva and Antonio Torralba. Building the gist of a scene: the role of global image features in recognition. *Progress in Brain Research*, 155:23–36, 2006.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Trans. Mach. Learn. Res.*, 2024.

Yunpeng Qu, Kaidong Zhang, Yukang Ding, Ying Chen, and Jian Wang. Semantic one-dimensional tokenizer for image reconstruction and generation. *CoRR*, abs/2603.16373, 2026.

Ali Razavi, Aäron van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with VQ-VAE-2. In *Advances in Neural Information Processing Systems*, 2019.

Oren Rippel, Michael A. Gelbart, and Ryan P. Adams. Learning ordered representations with nested dropout. In *International Conference on Machine Learning*, 2014.

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*, 2022.

Jürgen Schmidhuber. Reinforcement learning upside down: Don't predict rewards–just map them to actions. *CoRR*, abs/1912.02875, 2019.

Junhong Shen, Kushal Tirumala, Michihiro Yasunaga, Ishan Misra, Luke Zettlemoyer, Lili Yu, and Chunting Zhou. CAT: Content-adaptive image tokenization. In *Advances in Neural Information Processing Systems*, 2025.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seung Eun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *Trans. Mach. Learn. Res.*, 2026.

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

Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. In *Advances in Neural Information Processing Systems*, 2024.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *European Conference on Computer Vision*, 2020.

Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, 2017.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.*, 13(4):600–612, 2004.

Xin Wen, Bingchen Zhao, Ismail Elezi, Jiankang Deng, and Xiaojuan Qi. "Principal Components" enable a new language of images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

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

Jingfeng Yao, Bin Yang, and Xinggang Wang. Reconstruction vs. generation: Taming optimization dilemma in latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

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

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. In *International Conference on Learning Representations*, 2025.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Anlin Zheng, Xin Wen, Xuanyang Zhang, Chuofan Ma, Tiancai Wang, Gang Yu, Xiangyu Zhang, and Xiaojuan Qi. Vision foundation models as effective visual tokenizers for autoregressive generation. In *Advances in Neural Information Processing Systems*, 2025.

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

APPENDIX

TABLE OF CONTENTS

A Training and Evaluation Details **14**
A.1 Two-Stage Training 14
A.2 Hyperparameters 14
A.3 Evaluation Protocol 14

B Reproduction of the KARL Baseline **17**

C Comparison to Independently Trained Tokenizers **19**

D Tokens Used vs. Reconstruction Quality Metrics **21**

E Evaluation on Multi-Object Scenes (MS COCO) **22**

F Maximum Token Budget vs. Reconstruction Quality Metrics **24**

G Additional Qualitative KARL† vs. GIST Comparisons **26**
G.1 Synthetic and Abstract Stimuli 26
G.2 In-Distribution Samples (ImageNet-100) 27
G.3 Out-of-Distribution Natural Photographs 28
G.4 Difficult Cases: Textures, Fine Detail, and Noise 29

H Semantic Control: Qualitative Behavior **30**

I kNN Classification under Mean and Max Pooling **32**

J kNN Classification for Different Values of k **34**

K Attentive Probing **35**

A Training and Evaluation Details

This section documents the complete experimental setup: the two training stages (Appendix A.1), every hyperparameter of the models and of both stages (Appendix A.2), and the evaluation protocol (Appendix A.3).

A.1 Two-Stage Training

We train GIST in two sequential training stages that share the overall objective, but differ in the space in which $\mathcal{L}_{\text{rec}}$ is evaluated.

- **Stage 1 – Latent Distillation Pre-Training:** We distill a pretrained, frozen VQGAN (Esser et al., 2021) into the variable-length 1D GIST encoder. The GIST decoder predicts a VQGAN codebook distribution at every 2D grid position. Subsequently, $\mathcal{L}_{\text{rec}}$ is computed directly in the VQGAN token space using a cross-entropy over codebook indices and an MSE between predicted and ground-truth code embeddings. This avoids the costly pixel decoding during this pre-training stage.
- **Stage 2 – Pixel-Space Finetuning:** We finetune GIST jointly with the VQGAN to reconstruct high-fidelity images in pixel space. To achieve this, we pass the output of the GIST decoder into the VQGAN decoder to reconstruct the original image in pixel space. During this stage, $\mathcal{L}_{\text{rec}}$ operates in pixel space and comprises a pixel-wise ℓ_1 term, an LPIPS (Zhang et al., 2018) perceptual term, and an adversarial loss against a learned discriminator (Esser et al., 2021).

A.2 Hyperparameters

All models in this paper (KARL†, GIST, and GIST-1) are trained with the released KARL training script (Duggal et al., 2025a), which we leave unchanged except for the components that define our method: semantic conditioning and the frozen DINOv3 encoder (Siméoni et al., 2026) that measures the semantic error. Table 1 lists the architecture, data, and compute configuration of every model, and Table 2 the optimization settings of the two training stages of Appendix A.1.

Token budgets during training. In the first forward pass of Section 3.2, we draw the token budget T uniformly at random, with $T_{\min} = 16$ and $T_{\max} = 256$. The second forward pass always uses a budget of 256 tokens, i.e., $\Delta T = \max(0, 256 - T)$, and conditions the encoder on the realized errors ϵ^* after mapping each of them to the nearest bin that upper-bounds it.

A.3 Evaluation Protocol

Reconstruction. We compute all reconstruction metrics on the 5,000 images of the ImageNet-100 validation set at a 256×256 center crop, using the EMA weights of the final stage-2 checkpoint. In the token-budget comparisons (Appendices C and F), we evaluate the budgets $T \in \{32, 64, 96, 128, 160, 192, 224, 256\}$ with both targets fixed at 0. In the target-error comparisons (Figures 3 and 5), we fix $T = 256$ and vary each target over every bin of its dictionary: ϵ_{rec} for KARL†, the full $(\epsilon_{\text{rec}}, \epsilon_{\text{sem}})$ grid for GIST, and ϵ_{sem} for GIST-1. We count a token as active if its halting probability satisfies $\omega \leq \tau = 0.75$, and report the average number of active tokens over the validation set. We use linear interpolation between evaluated points to compute reported gains at matched mean active token counts. We use the AlexNet variant of LPIPS (Zhang et al., 2018), the default DreamSim ensemble (Fu et al., 2023), and SSIM (Wang et al., 2004) and PSNR on $[0, 1]$-scaled images.

kNN classification. As in Appendix I, we represent each image by the mean (or max) over its active register tokens of the 1024-d encoder outputs $\boldsymbol{Z}'_{\text{1D}}$ after the final LayerNorm, i.e., before the projection to the 12-d quantized latents. We use these features for both kNN classification and attentive probing to assess how semantic conditioning shapes the encoder representation, consistent with the encoder-level evaluation in ALIT (Duggal et al., 2025b). These evaluations measure the accessibility of class information before projection and quantization but do not directly assess its preservation in the discrete token sequence.

We ℓ_2-normalize these features and classify each query by an unweighted majority vote over its $k = 20$ nearest neighbors under cosine similarity, using the ImageNet-100 training set as the gallery and the validation set as queries, both at a 256×256 center crop and with $T = 256$. Appendix J repeats the comparison for $k \in \{10, 20, 100, 200\}$ and shows the ordering is unchanged.

Attentive probing. The probe of Appendix K, adopted from V-JEPA (Bardes et al., 2024), consists of a single learnable query that cross-attends to the frozen 1024-d encoder features used for the kNN evaluation (16 heads, MLP ratio 4, pre-LN), followed by a linear classifier over the 100 classes. We mask halted tokens ($\omega > 0.75$) out of this cross-attention, so the probe attends only over active tokens. For every target configuration and token budget, we train a separate probe for 10 epochs on the ImageNet-100 training set with AdamW (learning rate 10^{-3}, weight decay 0.01, batch size 64, cosine decay to 0 without warmup), a cross-entropy loss, and no data augmentation beyond the center crop. We report the best and last validation top-1 accuracy over the 10 epochs.

Table 1: **Model, data, and compute configuration of all models trained in this paper.** The cells spanning all columns apply to every model. The models differ only in which target errors they are conditioned on and in the frozen DINOv3 encoder of GIST and GIST-1.

Configuration	KARL†	GIST	GIST-1
Encoder / decoder depth	8 / 8		
Encoder / decoder width	1024 / 1024		
Attention heads	8		
MLP ratio	4		
Feedforward activation	GELU		
Trainable parameters	216M		
Image resolution	256^2		
Patch size	16×16 (256 patch tokens)		
Patch embedding	Convolutional patch features (768-d), concatenated with VQGAN encoder features (256-d)		
Halting MLP	Linear(1024, 512) – Tanh – Linear(512, 1)		
Halting threshold τ	0.75		
1D codebook	4096×12-d, ℓ_2-normalized		
Commitment cost	0.25		
Base 2D tokenizer	VQGAN (Esser et al., 2021)		
VQGAN checkpoint	MAGE ImageNet-1k release (Li et al., 2023)		
VQGAN downsampling factor	16		
VQGAN codebook	1024×256-d		
VQGAN parameters	54.5M		
Reconstruction embedding $\boldsymbol{E}_{\text{rec}}$	✓	✓	—
Semantic embedding $\boldsymbol{E}_{\text{sem}}$	—	✓	✓
Semantic encoder ϕ	—	DINOv3 ViT-L/16 (Siméoni et al., 2026)	
Semantic encoder parameters	—	303M (frozen)	
Semantic feature	—	Post-LN CLS token (1024-d)	
Dataset	ImageNet-100 (Tian et al., 2020)		
Training / evaluation images	≈127k / 5,000		
Augmentations	`RandomResizedCrop` (scale $[0.8, 1.0]$), `RandomHorizontalFlip`		
Evaluation preprocessing	256×256 center crop		
GPUs	8× NVIDIA H100 80GB		
Training cost	≈470 GPU-hours (≈2.5 days) per model		

Table 2: **Optimization settings of the two training stages.** Both stages use the released KARL recipe and are identical for KARL†, GIST, and GIST-1. Effective learning rates follow $\mathrm{lr} = \mathrm{base\ lr} \times \mathrm{batch\ size}/256$. Loss weights β and λ refer to the objective in Equation (10).

Configuration	Stage 1: latent distillation	Stage 2: pixel-space finetuning
Initialization	Random	Stage-1 EMA weights
Epochs	200	400
Warmup epochs	20	
Learning rate schedule	Linear warmup, cosine decay to 0	
Optimizer	AdamW	
Optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.95$	
Base learning rate	1×10^{-4}	
Effective learning rate	2×10^{-4}	5.6×10^{-5}
Batch size	512 (64 per GPU)	144 (18 per GPU)
Weight decay	0.05	
Gradient clipping norm	3.0	
Model EMA decay	0.9999	
Data type	`float16` autocast	
Reconstruction loss $\mathcal{L}_{\mathrm{rec}}$	Cross-entropy on VQGAN code indices + MSE to VQGAN code embeddings	L_1 + LPIPS-VGG (Zhang et al., 2018) + hinge adversarial loss
Loss weights β, λ	1, 1	
Adversarial loss weight	—	Adaptive (Esser et al., 2021), scaled by 0 (epochs 0–20), 0.2 (21–100), 0.8 (101–399)
Discriminator	—	PatchGAN (3 layers, 64 base channels)
Number of discriminators	—	8 (one per 32-token bucket of active tokens)
Discriminator optimizer	—	AdamW, same learning rate
Discriminator update	—	Every 9th iteration
VQGAN	Frozen	Finetuned (encoder and decoder)

B REPRODUCTION OF THE KARL BASELINE

In this paper, our primary baseline is KARL†, which denotes KARL retrained with its official, unmodified training code and the arguments of its released run scripts. We observe a small discrepancy between KARL's published numbers (Duggal et al., 2025a) and those we obtain by retraining the model with its unmodified training script. The discrepancy can originate on either (i) the *evaluation side*, if our protocol differs from the authors', or (ii) the *training side*, if the retrained model differs from the published checkpoint through code, configuration, hardware, or seed. To isolate the source of the discrepancy, we compare three configurations, all run through the same evaluation pipeline:

- **KARL (published).** The numbers reported in Table 1 of Duggal et al. (2025a).
- **KARL (public ckpt., our eval).** The official KARL checkpoint published on GitHub (`karl_small_vqgan_quantized_latents.pth`, trained on ImageNet-100).
- **KARL† (reproduction, our eval).** KARL retrained from the official, unmodified training script. We report three independent retrains to separate the training-side gap from seed variance.

We train on 8 H100 GPUs (two nodes of four) in the environment specified by the repository's `environment.yaml` (PyTorch 2.3.0, CUDA 12.1), with 8 processes, per-GPU batch sizes of 64 and 18, and 200 and 400 epochs for the latent-distillation and finetuning stages, respectively.

We summarize the results in Table 3, and draw three conclusions.

- **Our evaluation protocol reproduces the published numbers.** Row 2 matches row 1 to within 0.01 on every metric at every reported budget. Our evaluation protocol is thus not the source of the discrepancy.
- **The gap is on the training side, exceeds seed variance, and is concentrated in pixel-level metrics.** KARL† (rows 3–5) performs slightly worse than the released checkpoint (row 2) on L1 Loss $\times 10$ ($+0.04$ to $+0.10$) and SSIM (-0.02 to -0.04) at every budget; on LPIPS and DreamSim the gap is $+0.03$ at 32 tokens, at most $+0.02$ at 64 tokens, and within ± 0.01 from 128 tokens on. Across the three retrains, the spread between seeds is ≤ 0.02 on every metric and budget. This spread is far smaller than the gap to the published checkpoint, so the discrepancy is systematic rather than an artifact of seed variance. Its cause cannot be determined from the released artifacts: the published checkpoint stores only EMA weights and carries no record of the arguments, epoch count, or optimizer state that produced it.
- **The gap cancels in all controlled comparisons made in this paper.** Because KARL†, GIST, and GIST-1 share the same training and evaluation protocol, the gap introduced by the training script affects each method equally. The GIST and GIST-1 rows make this concrete: both match the KARL† mean to within 0.01 on every metric and budget, i.e., within seed spread, and sit below the published checkpoint by the same systematic gap. Our claims concern how the targets guide token allocation (Section 4), not base capacity, and are made relative to KARL† at matched base capacity. Comparing GIST against the published checkpoint would instead confound our contribution with an unexplained training difference. We therefore adopt KARL† as our controlled baseline.

Table 3: **Reproduction of the KARL baseline.** We compare KARL's published numbers (Duggal et al., 2025a) against the official checkpoint and KARL† (retrained from the official, unmodified code). Our evaluation of the official checkpoint (row 2) reproduces the published numbers (row 1) to within 0.01. The retrained models (rows 3–5) perform slightly worse than the official checkpoint, mainly on L1 and SSIM. Every std is at most 0.009, well below this gap, so the gap is systematic rather than an artifact of seed variance. The last two rows report GIST and GIST-1; both match KARL†. Every model trained from the released code thus shares the same base capacity, below the official checkpoint by the same gap.

	LPIPS ↓				DreamSim ↓				L1 Loss×10 ↓				SSIM ↑			
	32	64	128	256	32	64	128	256	32	64	128	256	32	64	128	256
KARL (published) (Duggal et al., 2025a)	**0.30**	—	**0.19**	**0.14**	**0.28**	—	0.16	**0.11**	**1.13**	—	**0.80**	**0.70**	**0.34**	—	0.44	**0.49**
KARL (public ckpt., our eval)	**0.30**	**0.24**	**0.19**	**0.14**	**0.28**	**0.21**	0.16	**0.11**	1.14	**0.98**	**0.80**	**0.70**	**0.34**	**0.38**	**0.45**	**0.49**
KARL† (reproduction, seed 1)	0.33	0.26	**0.19**	0.15	0.31	0.23	**0.15**	**0.11**	1.24	1.05	0.87	0.74	0.31	0.35	0.41	0.46
KARL† (reproduction, seed 2)	0.33	0.26	**0.19**	0.15	0.31	0.22	**0.15**	**0.11**	1.24	1.03	0.88	0.75	0.31	0.36	0.41	0.46
KARL† (reproduction, seed 3)	0.33	0.26	**0.19**	**0.14**	0.31	0.23	**0.15**	**0.11**	1.24	1.04	0.87	0.74	0.31	0.36	0.41	0.46
KARL† (reproduction, mean ± std)	0.33 ±.001	0.26 ±.002	**0.19** ±.001	0.15 ±.001	0.31 ±.002	0.22 ±.002	**0.15** ±.000	**0.11** ±.000	1.24 ±.003	1.04 ±.009	0.87 ±.007	0.74 ±.004	0.31 ±.001	0.36 ±.006	0.41 ±.002	0.46 ±.001
GIST	0.33	0.26	**0.19**	**0.14**	0.30	0.22	**0.15**	**0.11**	1.23	1.03	0.88	0.75	0.31	0.36	0.41	0.46
GIST-1	0.33	0.26	**0.19**	0.15	0.30	0.22	**0.15**	**0.11**	1.24	1.04	0.87	0.75	0.31	0.36	0.41	0.46

C Comparison to Independently Trained Tokenizers

In this section, we situate GIST relative to existing tokenizers, which differ from it on several axes at once: training data, scale, architecture, and objective. This comparison therefore serves to contextualize GIST rather than rank it against them. Because most existing tokenizers provide no comparable mechanism for controlling the encoding through a target error, we instead vary the *input token budget* T for each method, as in Appendix F.

We re-run each method in our own pipeline. To this end, we evaluate every method on the *same* ImageNet-100 validation set; only the *training data* differs (noted per row). For ALIT (a recurrent tokenizer that allocates 32 tokens per iteration), we use its reconstructions from rollout iterations $\{1, 2, 4, 8\}$, i.e. budgets $\{32, 64, 128, 256\}$, and disable its early-halting mechanism so that each budget uses its full token count. For FlexTok, we draw a single fixed-seed sample from its rectified-flow decoder with the default sampler (25 steps, default guidance scale); for One-D-Piece, we use its deterministic decoder.

The results are shown in Table 4. Within the controlled block, GIST and GIST-1 perform on par with KARL†: every difference is at most 0.01, i.e., within the seed spread of KARL†'s three retrains (Table 3). Furthermore, despite the reproduction gap detailed in Appendix B, GIST remains competitive with ALIT, the strongest independently trained tokenizer that does not share our baseline's method. For budgets ≥ 128 tokens, it matches or surpasses ALIT on L1, LPIPS, and SSIM (e.g. at 256 tokens, LPIPS 0.14 vs. 0.16 and SSIM 0.46 vs. 0.44).

Lastly, FlexTok and One-D-Piece generally underperform on LPIPS, L1, and SSIM compared to the other tokenizers. We attribute this gap to the metrics' focus on per-image reconstruction fidelity, which inherently favors reconstruction-trained tokenizers (GIST, KARL, and ALIT). In contrast, FlexTok and One-D-Piece trade pixel fidelity for generative realism by using a rectified-flow decoder and a perceptually trained decoder, respectively. Consistent with this trade-off, both methods remain competitive on DreamSim.

We omit rFID — the metric that would best capture this generative advantage — for two reasons. First, rFID is unpaired: it compares the *distribution* of reconstructions to that of the inputs, so a realistic reconstruction of the wrong instance (e.g. a plausible but different bird) is not penalized, whereas the per-image metrics above measure exactly this failure. Since our targets specify per-image error bounds, per-image fidelity is the quantity of interest; Duggal et al. (2025a) make the same argument for prioritizing per-image metrics over FID. Second, FID estimates are biased and high-variance at small sample sizes (Chong & Forsyth, 2020), and the 5,000 images of our ImageNet-100 validation set fall well below the regime where they become reliable. We note that Table 4 therefore does not reflect the generative realism that FlexTok and One-D-Piece are optimized for, for which we refer the reader to the rFID results in their respective papers (Bachmann et al., 2025; Miwa et al., 2025).

Table 4: **Reconstruction fidelity of adaptive tokenizers across token budgets on ImageNet-100.** Top block: models we train under one identical protocol (our controlled comparison); for KARL†, we report the mean over its three retrains (Table 3). Bottom block: Independently trained tokenizers, re-evaluated by us under the same protocol and reported for reference only. They differ from the controlled block in architecture, training objective, training data, and scale. In the top block, all methods perform comparably because GIST's advantage over the baselines comes only from varying its two targets, which are both fixed at 0 here. In the bottom block, the official KARL checkpoint performs best across most metrics.

		LPIPS ↓				DreamSim ↓				L1 Loss ×10 ↓				SSIM ↑			
Method	Train	32	64	128	256	32	64	128	256	32	64	128	256	32	64	128	256
Controlled: trained by us under identical conditions																	
KARL†	IN-100	**0.33**	**0.26**	**0.19**	0.15	0.31	**0.22**	**0.15**	**0.11**	1.24	1.04	**0.87**	**0.74**	**0.31**	**0.36**	**0.41**	**0.46**
GIST-1	IN-100	**0.33**	**0.26**	**0.19**	0.15	**0.30**	**0.22**	**0.15**	**0.11**	1.24	1.04	**0.87**	0.75	**0.31**	**0.36**	**0.41**	**0.46**
GIST	IN-100	**0.33**	**0.26**	**0.19**	**0.14**	**0.30**	**0.22**	**0.15**	**0.11**	**1.23**	**1.03**	0.88	0.75	**0.31**	**0.36**	**0.41**	**0.46**
Reference: independently trained — not directly comparable																	
KARL (public ckpt., our eval)	IN-100	**0.30**	**0.24**	**0.19**	**0.14**	0.28	0.21	0.16	0.11	**1.14**	**0.98**	**0.80**	**0.70**	**0.34**	**0.38**	**0.45**	**0.49**
ALIT (Duggal et al., 2025b)	IN-100	0.31	0.25	0.20	0.16	0.26	0.20	0.14	0.10	1.18	1.01	0.88	0.76	0.32	0.36	0.40	0.44
FlexTok-d12 (Bachmann et al., 2025)	IN-1K	0.42	0.35	0.27	0.21	0.27	0.22	0.17	0.14	1.53	1.33	1.05	0.85	0.27	0.30	0.38	0.45
FlexTok-d18/d18 (Bachmann et al., 2025)	IN-1K	0.37	0.31	0.25	0.19	**0.23**	0.19	0.16	0.12	1.37	1.21	1.02	0.84	0.30	0.33	0.38	0.45
FlexTok-d18/d28 (Bachmann et al., 2025)	IN-1K	0.38	0.35	0.27	0.20	0.24	0.22	0.17	0.12	1.39	1.31	1.05	0.84	0.29	0.31	0.37	0.45
One-D-Piece-S (Miwa et al., 2025)	IN-1K	0.36	0.28	0.21	0.19	0.29	0.20	0.14	0.12	1.35	1.17	1.02	0.94	0.32	0.35	0.38	0.40
One-D-Piece-B (Miwa et al., 2025)	IN-1K	0.35	0.27	0.21	0.18	0.27	0.18	0.13	0.11	1.35	1.17	1.01	0.92	0.32	0.35	0.38	0.41
One-D-Piece-L (Miwa et al., 2025)	IN-1K	0.35	0.26	0.20	0.17	0.25	**0.17**	**0.12**	**0.09**	1.33	1.14	0.96	0.88	0.33	0.35	0.39	0.41

D TOKENS USED VS. RECONSTRUCTION QUALITY METRICS

We complement the LPIPS and DreamSim Pareto fronts of Figure 3 with the full set of reconstruction quality metrics (L_1, LPIPS, SSIM, PSNR, and DreamSim) in Figure 7. Each panel compares GIST, GIST-1, and KARL† along the token-fidelity Pareto front in a different metric on the ImageNet-100 validation set, obtained by varying ϵ_{rec} over its full range for KARL† and GIST (fixing $\epsilon_{\text{sem}} = 0$), and by varying ϵ_{sem} for GIST-1. The conclusions drawn from Figure 3 hold across all metrics. GIST achieves better reconstruction quality than KARL† at any number of active tokens, and the gap widens as images are compressed into fewer tokens.

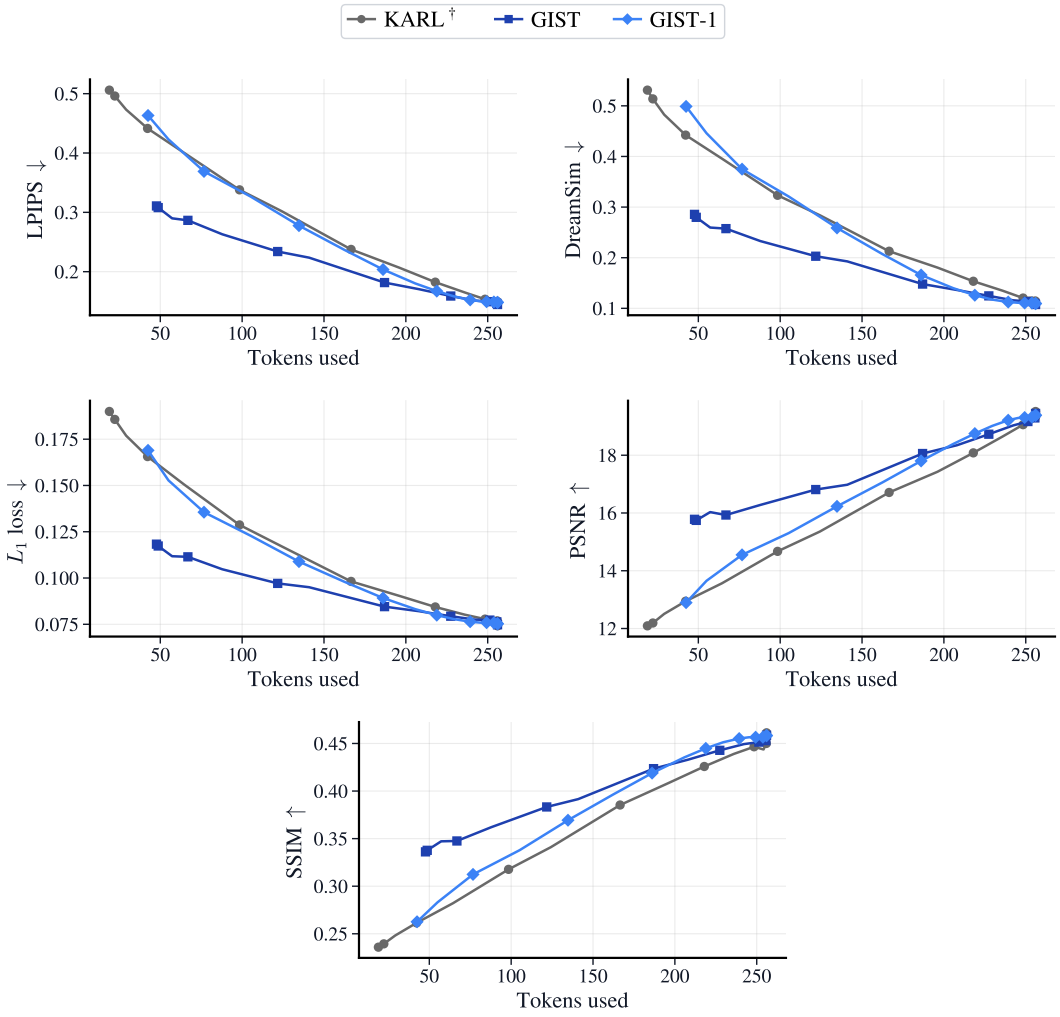


Figure 7: **Tokens vs. reconstruction quality metrics on ImageNet-100.** Pareto fronts on the ImageNet-100 validation set for the full set of reconstruction quality metrics (L_1, LPIPS, SSIM, PSNR, and DreamSim), obtained by varying ϵ_{rec} for KARL† and GIST (fixing $\epsilon_{\text{sem}} = 0$), and by varying ϵ_{sem} for GIST-1. GIST outperforms KARL† on every metric at any token count.

E Evaluation on Multi-Object Scenes (MS COCO)

To test whether the reconstruction gains extend beyond the object-centric images of ImageNet-100, we repeat the comparison of Figure 3 on MS COCO (Lin et al., 2014). Its everyday scenes contain multiple objects in context, allowing us to examine both generalization to a different image distribution and the effect of scene complexity.

Setup. We evaluate on the 5,000 images of the COCO 2017 validation set, which contain an average of 7.4 annotated object instances per image, with 87% containing at least two. All models use their ImageNet-100-trained checkpoints without further training or adaptation. To retain more scene context, we resize each image to 256 pixels on its shorter side and subsequently apply a 256×256 center crop. As in Figure 3, we fix the input token budget at $T = 256$ and vary ϵ_{rec} for KARL† and GIST (fixing $\epsilon_{\text{sem}} = 0$), and ϵ_{sem} for GIST-1. We report LPIPS and DreamSim at matched mean active token counts.

Reconstruction quality. Figure 8 shows that the findings on ImageNet-100 carry over to COCO. GIST achieves lower LPIPS and DreamSim than KARL† at every compared token count, with the largest relative gains under heavy compression: up to 27% lower LPIPS and 28% lower DreamSim. GIST-1 again performs on par with KARL†, suggesting that the reconstruction benefit of jointly controlling pixel-level and semantic fidelity extends to multi-object scenes outside the training distribution.

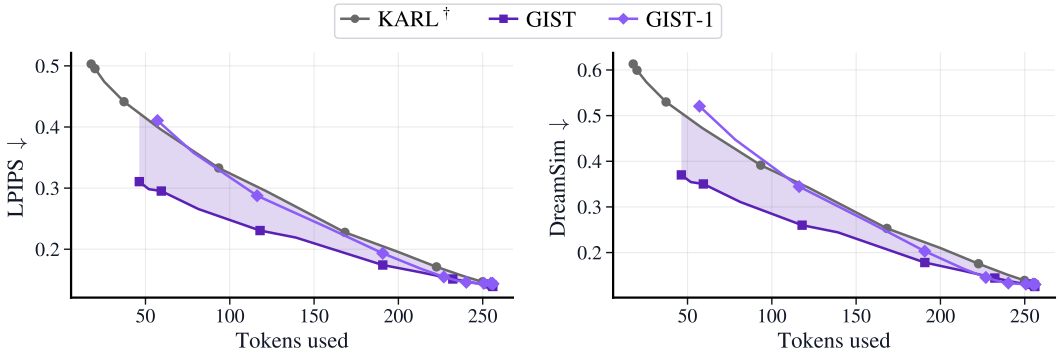


Figure 8: **The reconstruction gains extend to multi-object scenes.** Token-fidelity Pareto fronts on the COCO 2017 validation set, using the ImageNet-100-trained checkpoints without adaptation. As in Figure 3, we vary ϵ_{rec} for KARL† and GIST (fixing $\epsilon_{\text{sem}} = 0$), and vary ϵ_{sem} for GIST-1. The shaded regions highlight GIST's reduction in reconstruction error relative to KARL† at matched token counts.

Scene complexity. To examine how the gains vary with scene complexity, we group the validation images by their number of annotated object instances: 0–3, 4–8, and ≥ 9. These counts refer to the original images before cropping and serve as a proxy for scene complexity. We recompute the Pareto fronts within each group in Figure 9. At the same reconstruction target, both KARL† and GIST allocate more tokens to scenes containing more objects. For example, at $\epsilon_{\text{rec}} = 0.1$, KARL† uses an average of 102 tokens in the sparsest group and 154 in the densest.

GIST achieves lower LPIPS and DreamSim than KARL† at matched token counts in all three groups. The largest relative gains occur in the sparsest scenes: the maximum LPIPS reductions are 31%, 27%, and 22% from the sparsest to the densest group, with corresponding DreamSim reductions of 30%, 27%, and 24%. This difference is concentrated in the low-token regime. At 150 tokens, the LPIPS reduction is similar across groups (18–19%). Thus, GIST retains its advantage in crowded scenes, although under heavy compression the relative gains are smaller in scenes with more objects than in those with fewer.

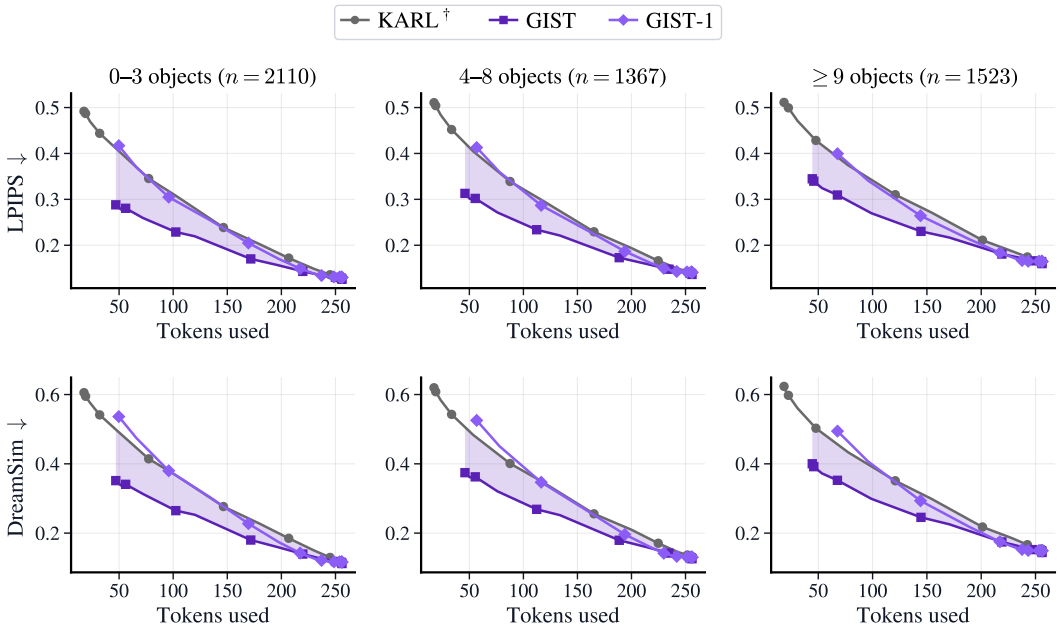


Figure 9: **The reconstruction advantage persists in crowded scenes.** Same evaluation as Figure 8, with images grouped by the number of annotated object instances in the original image. Each column corresponds to an object-count range, and n denotes the number of images in that group. GIST outperforms KARL† in every group, with the largest relative gains in sparse scenes at low token counts.

F MAXIMUM TOKEN BUDGET VS. RECONSTRUCTION QUALITY METRICS

The comparisons in Figure 3 and Appendix D vary the target errors and let each model decide how many tokens to halt and which ones. In this section, we complement these comparisons by holding the targets fixed and varying the input token budget T instead. This isolates the capacity of the underlying tokenizer from the effects of varying its targets.

In Figure 10, we compare GIST, GIST-1, and KARL† on the ImageNet-100 validation set across all five reconstruction quality metrics (L_1, LPIPS, SSIM, PSNR, DreamSim) as we vary the input token budget T with all targets held fixed at zero. GIST, GIST-1, and KARL† perform on par on every metric and at every token budget. This stands in contrast to the comparison under varying targets, where GIST Pareto-improves reconstruction quality over KARL† on every metric.

We interpret these results as follows. All three models share the same architecture, the same VQGAN base, and the same training recipe; they differ only in the target errors that condition the encoder. Varying T with the targets held fixed thus probes the capacity of the base tokenizer, which semantic conditioning leaves unchanged. What semantic conditioning changes is how the tokenizer models token complexity. KARL† models complexity along a single reconstructive axis: it uses a pixel-wise error target to decide how many tokens to halt and which ones. GIST-1 replaces this axis with a semantic one, and GIST models complexity along both. These axes determine how a model halts tokens as its targets are relaxed. Under a purely reconstructive axis, tokens are halted by pixel-level redundancy alone, which is blind to the spatial structure that carries an image's meaning. Under both axes, the semantic target can instead act as a guardrail that protects the image's semantic identity as ϵ_{rec} is relaxed (Figure 4). The gains of GIST under varying targets therefore stem from how conditioning on these targets organizes the latent space along complementary axes of complexity, rather than from base tokenizer capacity.

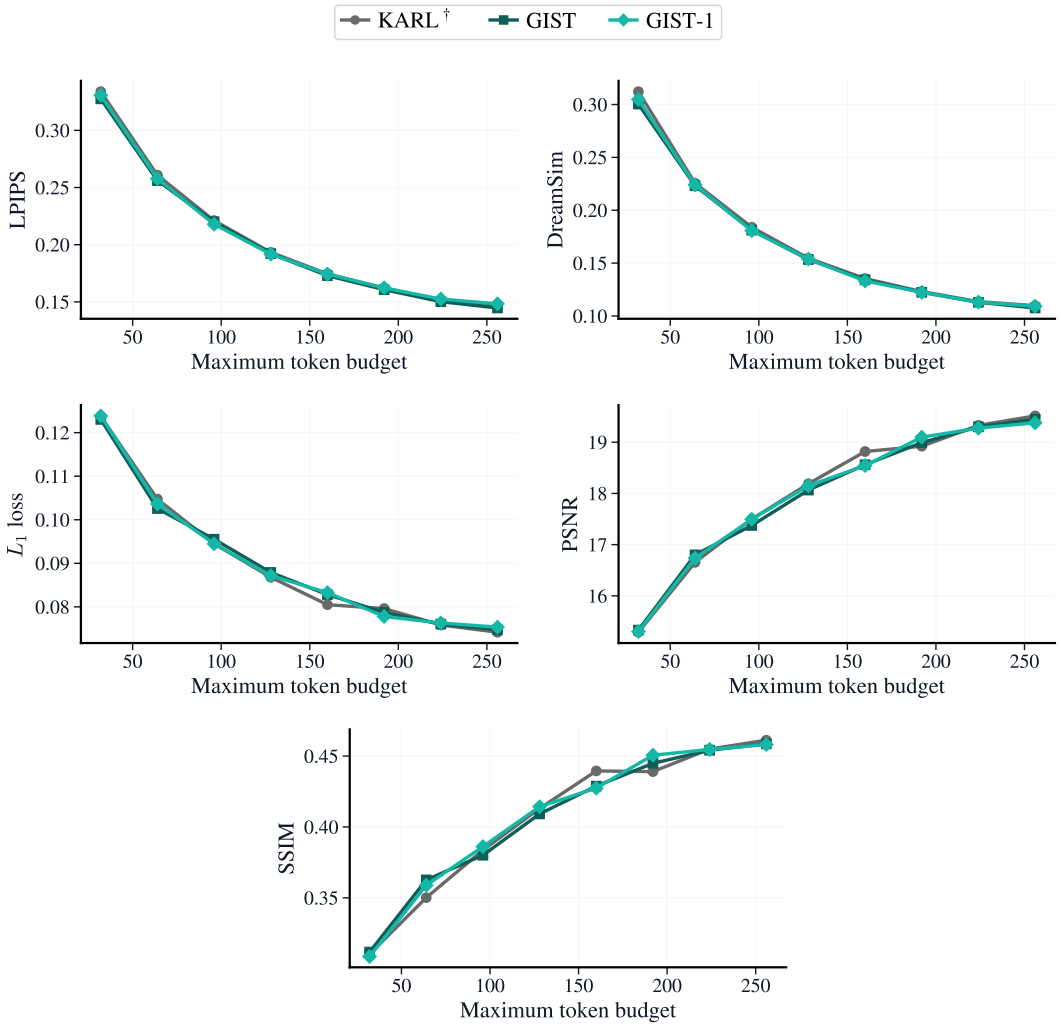


Figure 10: **Reconstruction quality metrics vs. input token budget on ImageNet-100.** We compare GIST, GIST-1, and KARL† across all five reconstruction quality metrics (L_1, LPIPS, SSIM, PSNR, DreamSim) on the ImageNet-100 validation set as we vary the input token budget T, with all targets fixed at zero. The three models perform on par on every metric, in contrast to the comparison under varying targets (Figures 3 and 7), where GIST Pareto-improves reconstruction quality over KARL†.

G Additional Qualitative KARL† vs. GIST Comparisons

We complement the two qualitative examples in the main paper with additional reconstructions from KARL† and GIST (fixing $\epsilon_{\text{sem}} = 0$) at $T = 256$ tokens, organized by image complexity.

G.1 Synthetic and Abstract Stimuli

Synthetic inputs isolate frequency content and color structure without natural-image semantics.

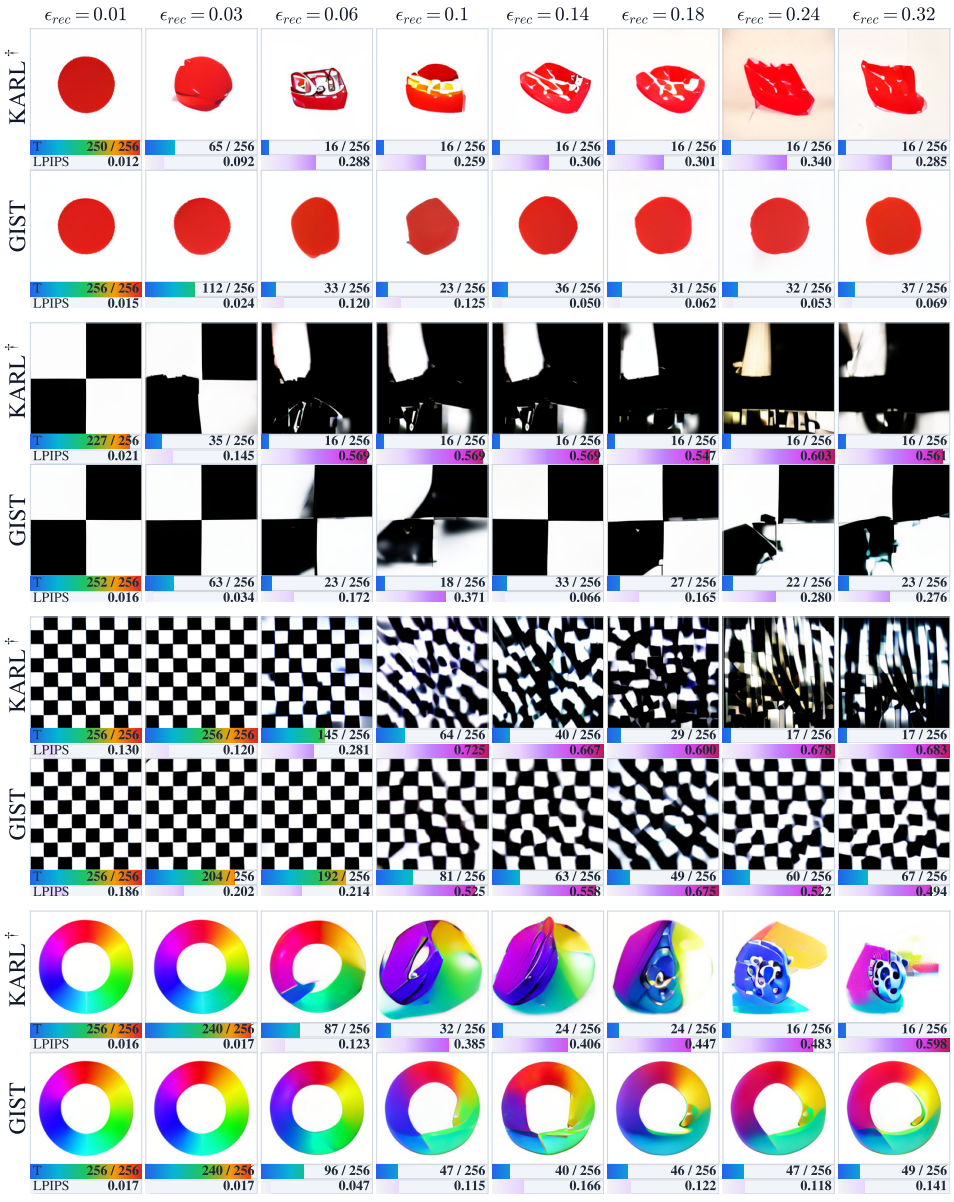


Figure 11: **Synthetic and abstract stimuli.** KARL† (top of each pair) vs. GIST (bottom of each pair) reconstructions across the ϵ_{rec} spectrum. From top to bottom: *red circle on white*, *checkerboard* 2×2, *checkerboard* 8×8, and *color wheel*.

G.2 In-Distribution Samples (ImageNet-100)

We use natural images drawn from classes that lie inside the training distribution. These probe the typical operating regime of both tokenizers.

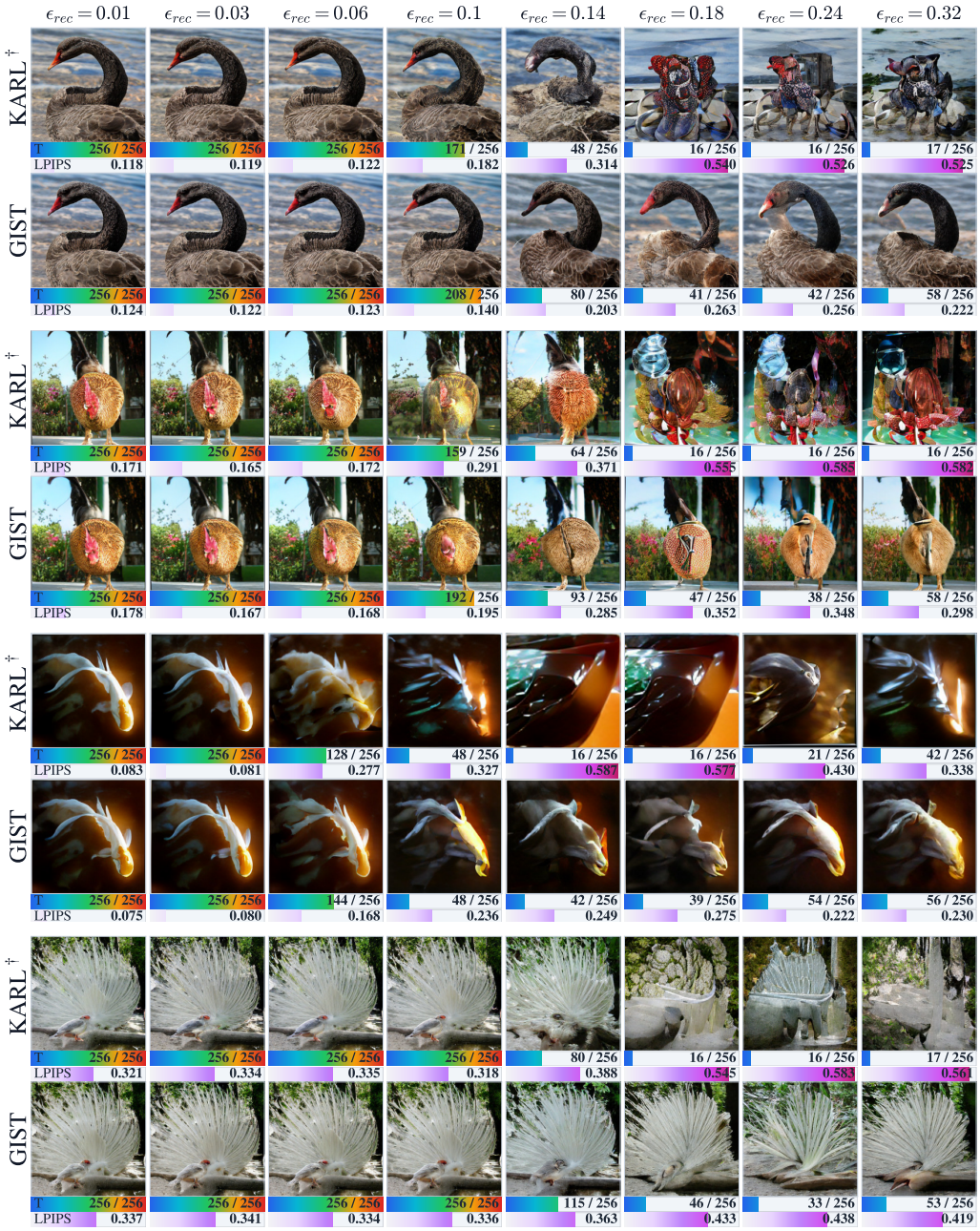


Figure 12: **In-distribution samples (ImageNet-100).** Same setup as Figure 11. From top to bottom: *black swan*, *rooster*, *goldfish*, and *peacock*.

G.3 Out-of-Distribution Natural Photographs

We use real-world photographs of subjects whose classes are not part of the ImageNet-100 training set. They remain natural-image content, but their specific categories and compositional context lie outside the training distribution.

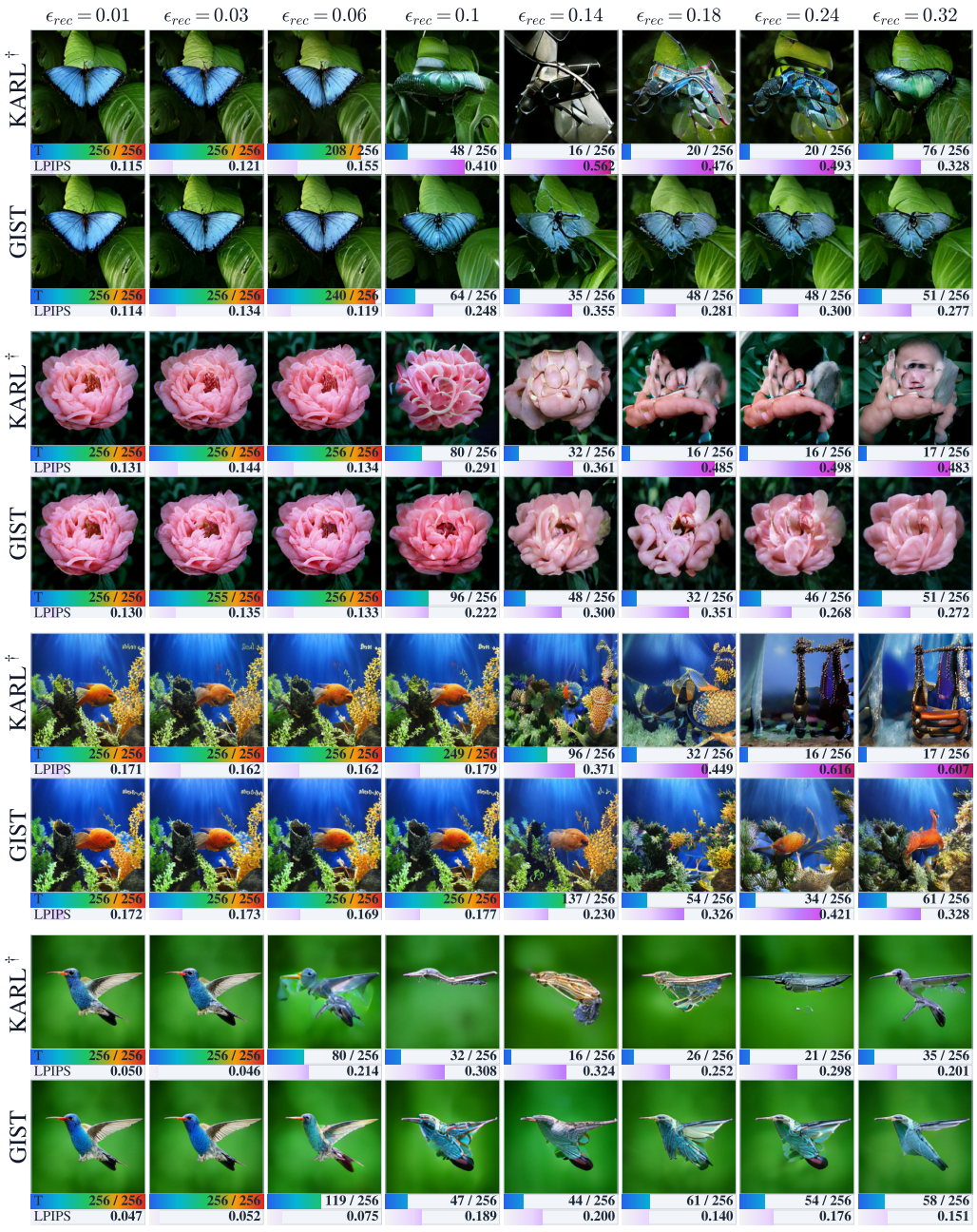


Figure 13: **Out-of-distribution natural photographs.** Same setup as Figure 11. From top to bottom: *blue morpho butterfly*, *pink peony*, *goldfish reef*, and *hummingbird*.

G.4 Difficult Cases: Textures, Fine Detail, and Noise

Textures, fractals, dense text, and low-frequency noise are dominated by fine-grained detail spread across the entire image rather than by a distinct subject. Such content is costly to compress, and it probes how both tokenizers behave when no single subject carries the image's identity.

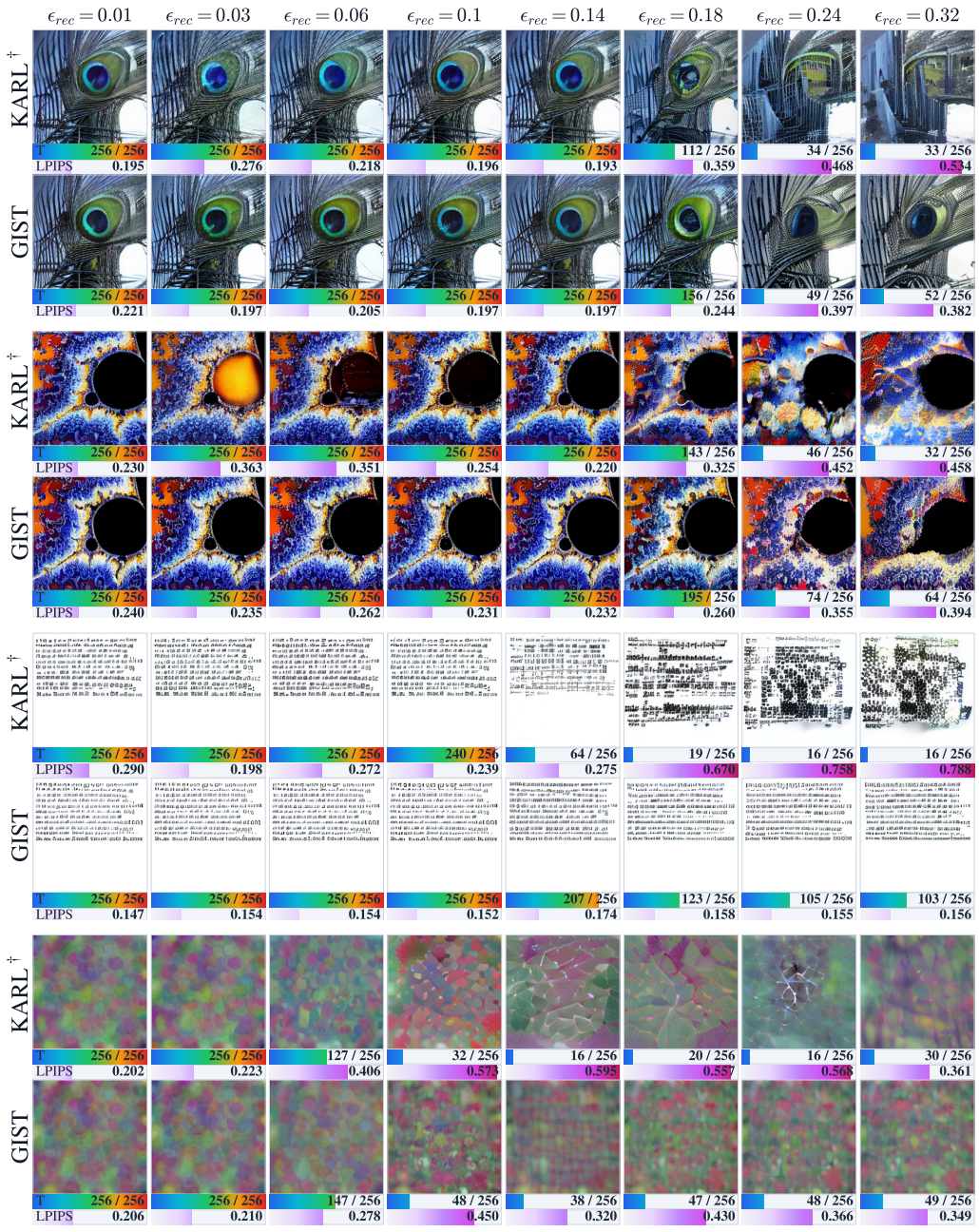


Figure 14: **Difficult cases: textures and high-frequency detail.** Same setup as Figure 11. From top to bottom: *peacock feather*, *Mandelbrot fractal*, *text paragraph*, and *low-frequency noise*.

H SEMANTIC CONTROL: QUALITATIVE BEHAVIOR

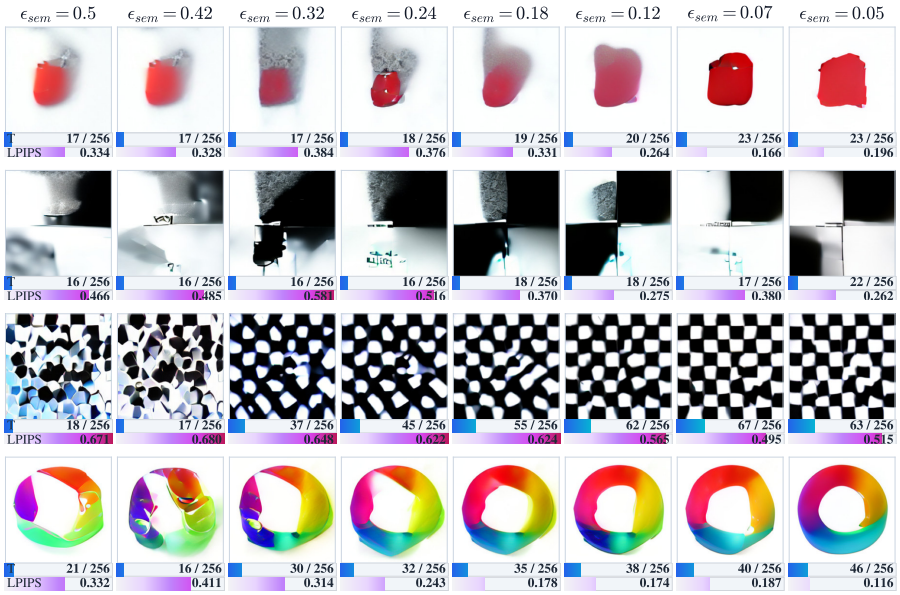


Figure 15: **Synthetic and abstract stimuli.** GIST reconstructions while varying ϵ_{sem}, with $\epsilon_{\text{rec}} = 0.24$ and $T = 256$ tokens. From top to bottom: *red circle on white*, *checkerboard* 2×2, *checkerboard* 8×8, and *color wheel*.

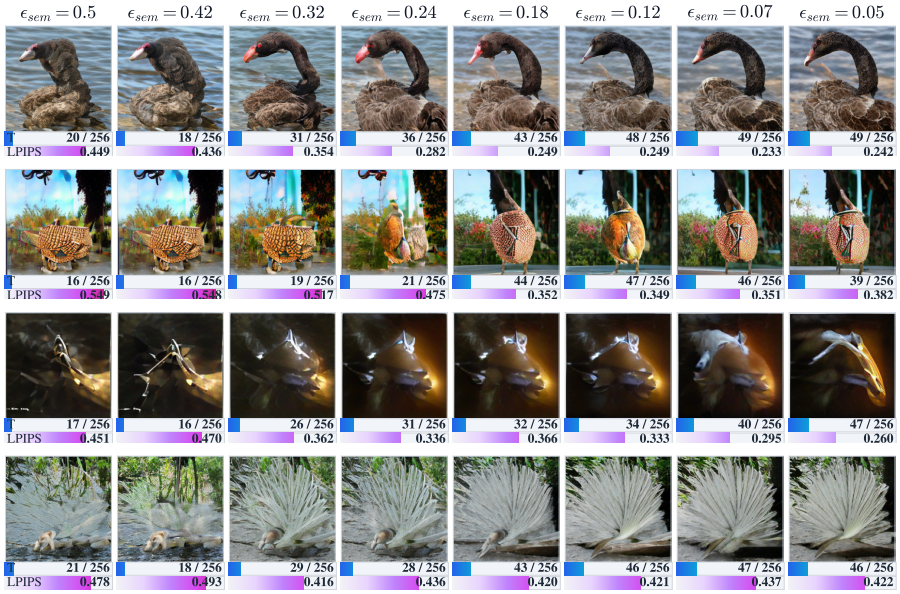


Figure 16: **In-distribution samples (ImageNet-100).** Same setup as Figure 15. From top to bottom: *black swan*, *rooster*, *goldfish*, and *peacock*.

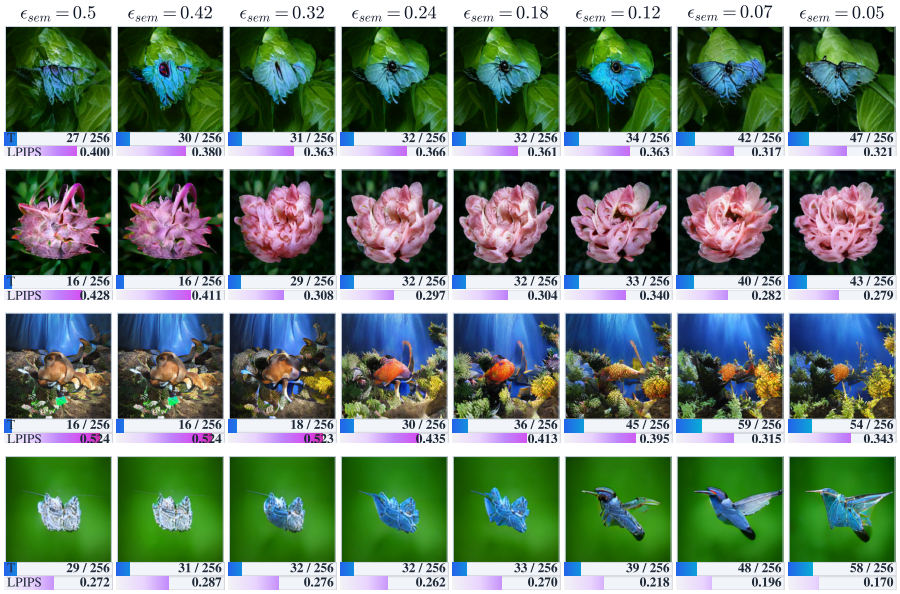


Figure 17: **Out-of-distribution natural photographs.** Same setup as Figure 15. From top to bottom: *blue morpho butterfly*, *pink peony*, *goldfish reef*, and *hummingbird*.

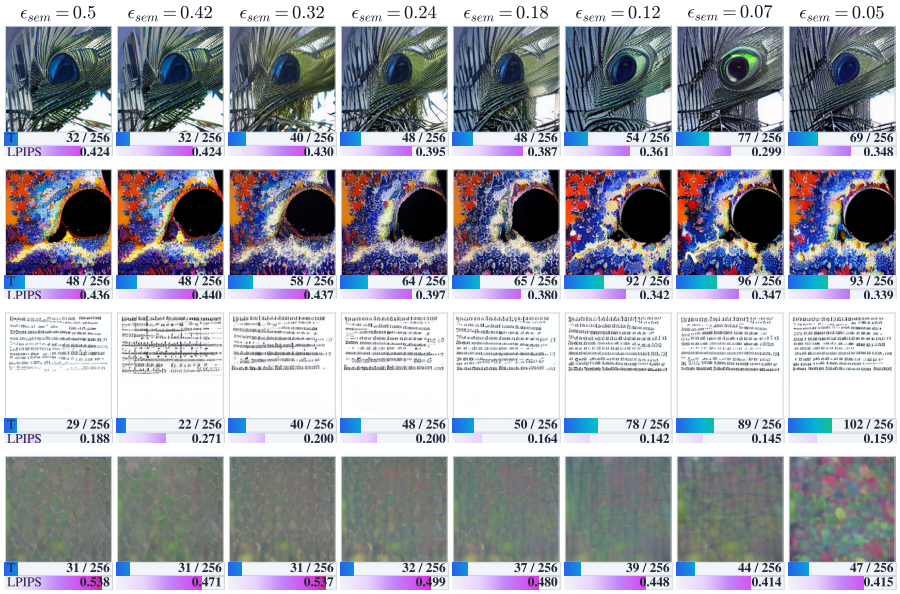


Figure 18: **Difficult cases: textures and high-frequency detail.** Same setup as Figure 15. From top to bottom: *peacock feather*, *Mandelbrot fractal*, *text paragraph*, and *low-frequency noise*.

I kNN Classification under Mean and Max Pooling

We measure the semantic informativeness of the latent tokens with a kNN classifier (k=20) on ImageNet-100, using the training set as gallery and the validation set as queries. Since the number of active tokens varies per image, we pool the active register tokens of each image into a single vector. In the main paper, we use mean pooling (Figure 6a). In this appendix, we additionally report max pooling.

The two poolings summarize the tokens differently. Mean pooling averages over all active tokens, so every token contributes equally to the representation. Tokens that carry semantic information add to the class signal, whereas tokens that carry little semantic information dilute it. Max pooling instead retains, for each feature dimension, the largest activation across the active tokens. A token thus affects the representation only in the dimensions where it exceeds all other tokens. Max pooling is therefore less sensitive than mean pooling to the addition of tokens that carry little semantic information. A higher accuracy under max pooling than under mean pooling hence suggests that some tokens dilute the semantic information carried by the others.

In Figure 19, we compare the kNN accuracy of GIST, GIST-1, and KARL† under mean and max pooling as we vary each method's target values (same setup as Figure 6a). We make two observations. First, the ordering of the three methods is unchanged under max pooling. GIST-1 performs best, followed by GIST and KARL†, at any number of active tokens. The conclusions of the main paper therefore do not depend on the choice of pooling.

Second, the relation between the two poolings differs across methods. For GIST-1, mean pooling yields a higher accuracy than max pooling, and accuracy increases approximately linearly with the number of active tokens. For GIST and KARL†, we observe the opposite. Max pooling yields a slightly higher accuracy than mean pooling, and accuracy remains approximately constant as the number of active tokens increases. This difference corresponds to the target varied for each method.

- The curve of GIST-1 is obtained by varying ϵ_{sem}, whereas the curves of GIST and KARL† are obtained by varying ϵ_{rec} (with $\epsilon_{\text{sem}} = 0$ for GIST). Lowering ϵ_{sem} demands tokens that preserve what the image depicts. The tokens added along this sweep therefore carry semantic information, and accuracy increases with their number. Under mean pooling, each of these tokens contributes to the representation, whereas under max pooling a token contributes only in the dimensions where it exceeds the others. Mean pooling can therefore benefit more from the added tokens, which is consistent with its higher accuracy for GIST-1.
- Lowering ϵ_{rec} instead demands tokens that reproduce the exact pixels of the image. The tokens added along this sweep encode pixel-level detail and carry little class information. Under mean pooling, these tokens dilute the semantic signal of the remaining tokens, whereas under max pooling they leave the largest activations largely unchanged. This is consistent with the slightly higher accuracy of max pooling for GIST and KARL†, and with their accuracy remaining approximately constant.

We note that these differences are subtle, and we do not draw strong conclusions from them. They are, however, consistent with Figure 6b and with the view that the targets control not only how many tokens an image requires, but also what those tokens encode.

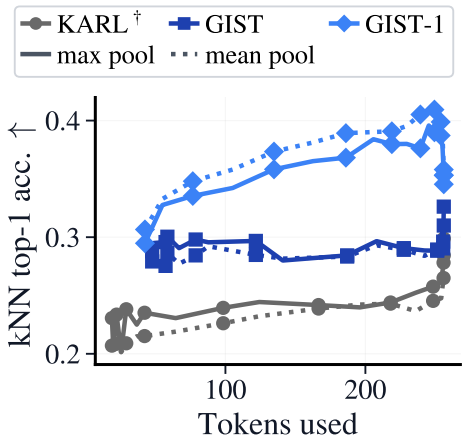


Figure 19: **The kNN comparison is robust to the choice of pooling.** kNN accuracy of GIST, KARL†, and GIST-1 on the ImageNet-100 validation set, obtained by varying ϵ_{rec} for KARL† and GIST (fixing $\epsilon_{\text{sem}} = 0$), and by varying ϵ_{sem} for GIST-1 (same setup as Figure 6a). Accuracies are computed by max pooling (solid) and mean pooling (dotted) over the active tokens. The ordering of the three methods is the same under both poolings. For GIST-1, mean pooling yields a higher accuracy, and accuracy increases with the number of active tokens. For GIST and KARL†, max pooling yields a slightly higher accuracy, and accuracy remains approximately constant.

J KNN CLASSIFICATION FOR DIFFERENT VALUES OF k

In the main paper, we report kNN accuracy with k=20 neighbors (Figure 6a). In Figure 20, we additionally report $k \in \{10, 100, 200\}$, with all other settings unchanged: mean pooling over the active tokens, the ImageNet-100 training set as gallery, and an unweighted majority vote over the retrieved neighbors. The ordering of the three methods is the same for every k. GIST-1 performs best, followed by GIST and KARL†, at any number of active tokens, and its advantage over KARL† lies between 6 and 17 points throughout. The conclusions of the main paper therefore do not depend on the choice of k.

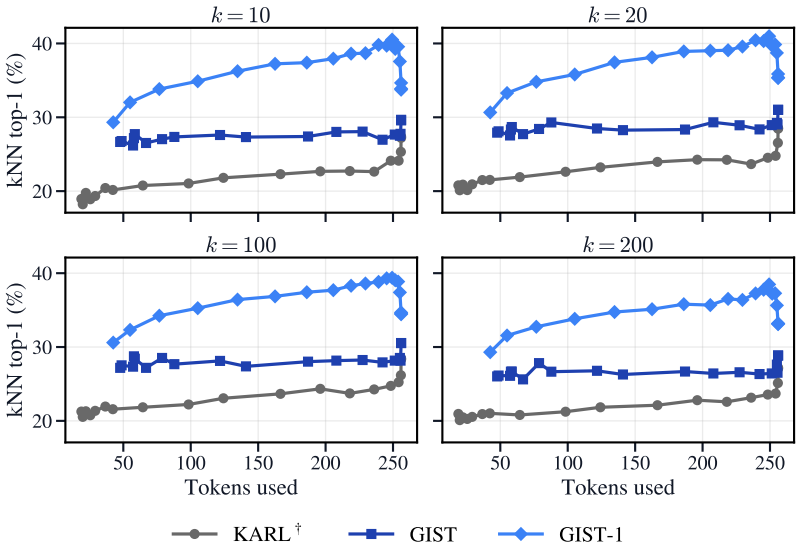


Figure 20: **The kNN comparison is robust to the choice of k.** kNN accuracy of GIST, KARL†, and GIST-1 on the ImageNet-100 validation set for $k \in \{10, 20, 100, 200\}$ (same setup as Figure 6a). The ordering of the three methods is the same for every k: GIST-1 performs best, followed by GIST and KARL†, at any number of active tokens. Accuracy is highest at k=20 for all three methods and decreases slightly at k=200.

Table 5 reports the mean accuracy over the range of active tokens shared by all three methods, together with the advantage of GIST and GIST-1 over KARL†. The mean advantage varies little with k: it is 12.6–14.3 points for GIST-1 and 4.5–5.4 points for GIST. Both advantages are largest at k=10 and decrease slightly as k grows.

Table 5: **Mean kNN accuracy of KARL†, GIST, and GIST-1 for different values of k.** Mean kNN accuracy of KARL†, GIST, and GIST-1 on the ImageNet-100 validation set for $k \in \{10, 20, 100, 200\}$ (same setup as Figure 6a), averaged over the range of active tokens shared by all three methods (47.7–256 tokens). The last two columns report the advantage of GIST and GIST-1 over KARL† in mean accuracy. For every k, GIST-1 performs best, followed by GIST and KARL†. The advantage of GIST-1 over KARL† lies between 12.6 and 14.3 points, and that of GIST over KARL† between 4.5 and 5.4 points. Both advantages are largest at k=10 and decrease slightly as k grows.

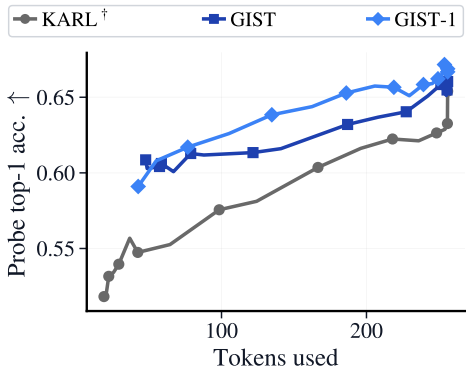
	Mean kNN top-1 (%)			Advantage over KARL†	
k	KARL†	GIST	GIST-1	GIST	GIST-1
10	22.0	27.4	36.3	+5.4	+14.3
20	23.4	28.6	37.3	+5.2	+13.9
100	23.3	27.9	36.3	+4.6	+13.0
200	22.1	26.6	34.7	+4.5	+12.6

K ATTENTIVE PROBING

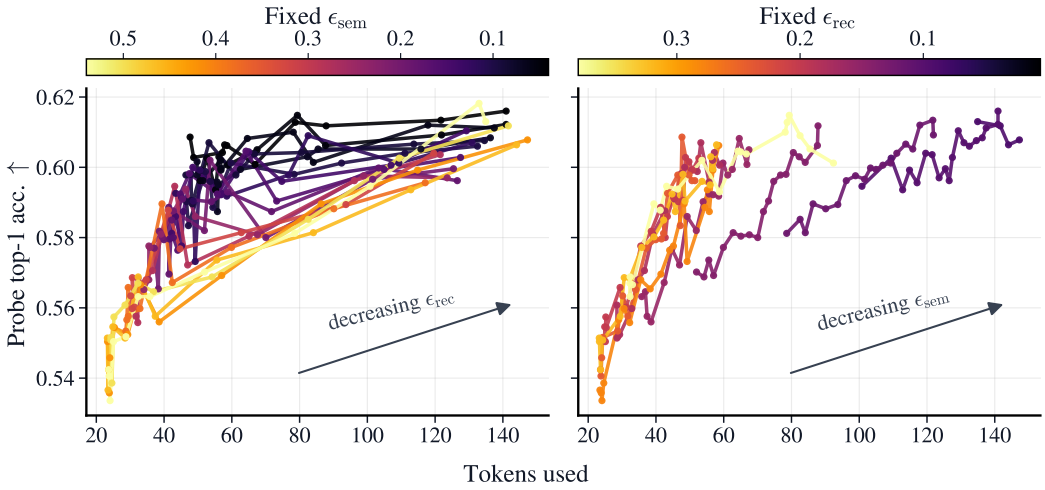
To complement the parameter-free kNN evaluation in Figure 6a, we probe the same frozen tokens with a trained classifier. Specifically, for every target configuration of every tokenizer, we train a separate lightweight attentive probe, in which a learnable query cross-attends to the image's active tokens (i.e., with halted tokens masked out), and a linear classifier maps the resulting vector to class logits. Architecture and training details of the probe are given in Appendix A.3. We train the probe on the ImageNet-100 training set and report top-1 accuracy on the validation set for the same target values as in Figure 6a.

In Figure 21a, we compare the attentive-probe accuracy of GIST, GIST-1, and KARL† as we vary each method's target values. The ordering from the kNN evaluation is preserved (GIST-1 performs best, followed by GIST and KARL†), but the gaps narrow. GIST-1 and GIST lead KARL† by $+6$ and $+5$ points at 64 tokens and by $+4$ and $+2$ points at 192 tokens, compared to $+12$ and $+6$ points at 64 tokens and $+15$ and $+4$ points at 192 tokens under kNN (Figure 6a). We draw two conclusions. First, the narrower gaps show that a considerable part of the kNN advantage lies in how *readily* semantic information can be read out. A kNN classifier only sees the class information that mean pooling exposes directly, whereas a trained probe can attend to the relevant tokens and combine them. Second, the gaps do not close. In other words, even a probe that is free to select and weight tokens recovers more class-relevant information from GIST and GIST-1 tokens than from KARL† tokens. Semantic control therefore not only makes semantic information more accessible, but also increases how much of it the tokens carry.

In Figure 21b, we repeat the single-target trajectories of Figure 6b with probe accuracy in place of kNN accuracy. The picture mirrors the main paper. Lowering ϵ_{rec} (left) yields long, shallow curves along which accuracy grows only slowly with the number of tokens, whereas lowering ϵ_{sem} (right) yields short, steep curves that gain accuracy over only a few added tokens. Lowering the semantic target is thus the more token-efficient way to make tokens semantically informative, whether the readout is a nearest-neighbor lookup or a trained probe.



(a) GIST vs. KARL† vs. GIST-1



(b) Varying one target at a time

Figure 21: **The advantage of semantic control holds under a trained attentive probe.** Both panels report the best validation top-1 accuracy over the 10 training epochs for each target configuration. (a) Attentive-probe accuracy of GIST, KARL†, and GIST-1 as we vary each method's target values (same setup as Figure 6a). The ordering from the kNN evaluation is preserved, with narrower but consistent gaps: GIST-1 and GIST outperform KARL† at every token count. (b) Attentive-probe accuracy trajectories of GIST as we vary one target with the other held fixed (shown by color). Lowering ϵ_{sem} increases probe accuracy at a much steeper rate than lowering ϵ_{rec}. Together, the probe shows that conditioning on ϵ_{sem} not only exposes semantic information more readily, but also makes the tokens carry more of it.

Table 6 compares best-epoch and final-epoch accuracies at 64 and 192 tokens. The ordering is unchanged, and the advantages of GIST and GIST-1 over KARL† change by at most 0.8 percentage points.

Table 6: **Best-epoch vs. final-epoch attentive-probe accuracy.** Top-1 accuracy (%) at 64 and 192 tokens, obtained by linear interpolation between neighboring target configurations on each method's curve. For each configuration, *Best epoch* uses the highest validation accuracy over the 10 training epochs, as in Figure 21a, while *Final epoch* uses the validation accuracy after epoch 10.

Method	Tokens	Best epoch	Final epoch
KARL†	64	55.2	54.8
GIST	64	60.3	59.9
GIST-1	64	61.2	60.5
KARL†	192	61.4	60.6
GIST	192	63.3	63.1
GIST-1	192	65.4	65.4