arXiv:2602.16086v4 [cs.CV] 6 Sep 2026

KURBAN INTELLIGENCE LAB

PREPRINT

Under review

LGQ: Learnable Geometric Quantization for Image Tokenization

A learnable codebook that never collapses: annealed soft assignment with a K-scaled diversity regularizer

Idil Bilge Altun[1] · Mert Onur Cakiroglu[1] · Elham Buxton[2] · Mehmet Dalkilic[1] · Hasan Kurban[3,✉]

[1] Luddy School of Informatics, Computing, and Engineering, Indiana University Bloomington, Bloomington, IN, USA
[2] Department of Computer Science, University of Illinois Springfield, Springfield, IL, USA
[3] College of Science and Engineering, Hamad Bin Khalifa University, Doha, Qatar

✉ Correspondence: hkurban@hbku.edu.qa

ABSTRACT

Recent collapse-free quantizers such as FSQ achieve stable training by replacing the learnable codebook with an engineered geometry: a fixed scalar grid whose structure is dictated by the codebook size K. We show this trade-off is unnecessary. We introduce **Learnable Geometric Quantization (LGQ)**, which retains a learnable codebook of codes and performs soft-to-hard assignment via temperature annealing, regularized by two cheap terms: a *diversity* term scaled by codebook size that penalizes concentrated batch-average usage is the primary driver of collapse resistance, complemented by a *peakedness* term that sharpens each token's soft-assignment toward one-hot; together they prevent codebook collapse without EMA, reset heuristics, or codebook reparameterization. Under a fixed VQ-GAN backbone, we benchmark LGQ against RotVQ, FSQ, LFQ, SimVQ, and IBQ on ImageNet 256×256 at $K = 16{,}384$, and sweep LGQ over $K \in \{4096, \ldots, 65{,}536\}$ without any per-K hyperparameter tuning. LGQ attains the best reconstruction FID at $K = 16{,}384$ while maintaining 100% codebook utilization, and continues to improve as the codebook grows to $K = 65{,}536$, holding 100% utilization at every K. Training MaskGIT on the frozen tokenizers, LGQ further attains the best class-conditional generation among the compared quantizers, leading on reconstruction and generation alike. Code is available at https://anonymous.4open.science/r/lgq-anon-E12C/.

KEYWORDS image tokenization; vector quantization; codebook collapse; discrete representation learning; generative models

1 Introduction

Discrete representation learning underpins modern generative image pipelines: vector-quantized autoencoders [1, 2] compress an image into a short sequence of tokens that autoregressive [1, 3]

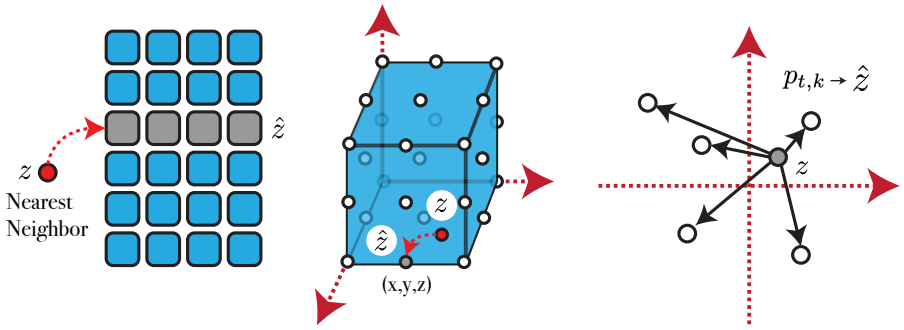


Figure 1 | VQ-VAE, FSQ and LGQ comparison. Conceptual comparison of three discrete quantizers (VQ, FSQ, LGQ). VQ's flat learned codebook collapses; FSQ's fixed grid is collapse-free but unable to adapt to the data anisotropy; LGQ keeps a *learnable* codebook with explicit usage regularization.

and masked-generative [3, 4] models then learn to predict. This makes the tokenizer the ceiling of the entire system. No downstream model can recover detail its tokenizer has discarded, and a degenerate token distribution, one where a handful of codes absorb most of the usage, leaves the generator with a vocabulary that is effectively far smaller than it appears. The central obstacle to a good tokenizer is therefore keeping a large codebook fully and meaningfully used, the problem this paper addresses.

Three classes of quantizer dominate the recent literature (Fig. 1). (i) **Vector Quantization (VQ)** [1] learns a flat codebook updated by exponential moving averages, but is widely reported to undergo *codebook collapse* [2, 5]: a small fraction of entries dominate. (ii) **Finite Scalar Quantization (FSQ)** [6] replaces the learned codebook with fixed per-channel scalar grids, eliminating collapse by construction at the cost of flexibility. (iii) **Lookup-Free Quantization (LFQ)** [4] encodes each spatial location as a binary string, making the codebook implicit. Two more recent designs like SimVQ [5] that reparameterizes the codebook through a learned linear projection, and the rotation trick [7] that transforms gradients through quantization attempt to fix VQ's optimization pathology directly. A common thread runs through the recent collapse-resistant designs: they either (a) restrict where codes may be placed (FSQ, LFQ rigidly, SimVQ and VQGAN-LC softly), (b) restrict how many codes receive gradient by assigning hard (VQ, RotVQ), or (c) change the rule by which codes are scored (IBQ). LGQ instead maintains a fully learnable codebook and pays for collapse resistance with two cheap soft-assignment regularizers.

Contributions.

- We introduce **LGQ**, a learnable quantizer whose soft-assignment, temperature schedule, and K-scaled diversity regularizer together eliminate codebook collapse with a single fixed hyperparameter setting across a 16× codebook-size range (K=4K to K=65K) without EMA, reset, reparameterization, or per-scale structural redesign, holding 100% utilization at every K where IBQ under the identical protocol falls to 55% at 65K (Secs. 3 and 7.2).
- We provide a controlled $K = 16{,}384$ **comparison** of six quantizers on ImageNet 256×256 under an identical VQ-GAN-style backbone and reconstruction objective, in which LGQ achieves the best rFID, PSNR, SSIM, and LPIPS (Sec. 7.1).

- We **ablate** the regularizer weights (λ_{peak}, λ_{div}) and the temperature schedule (τ_{start}, τ_{end}, annealing curve), isolating the design choices that drive LGQ's collapse resistance (Sec. A), and isolate the selection rule with a matched logit ablation (Sec. 5.2).
- We report a **generation-side comparison**: a small MaskGIT transformer trained on each frozen tokenizer under a matched recipe, on which LGQ achieves the best class-conditional gFID among the five tokenizers (Table 5).

2 Background

Discrete tokenizers. A discrete tokenizer encodes an image as a short sequence of tokens from a fixed vocabulary, so that discrete sequence models, e.g., autoregressive transformers or MaskGIT [3], can be trained to generate images. It consists of an encoder, quantizer, and decoder (E, Q, D). The encoder produces a grid of continuous embeddings $\mathbf{z} = E(x) \in \mathbb{R}^{B\times T\times C}$ (B batch size, T spatial tokens, C embedding dimension); for a single token we write $\mathbf{z} \in \mathbb{R}^C$. The quantizer holds a *codebook* $\mathcal{C} = \{\mathbf{e}_k\}_{k=1}^K$ and maps each token to an index $k = Q(\mathbf{z}) \in \{1, \dots, K\}$ with embedding $\hat{\mathbf{z}} = \mathbf{e}_k$; the decoder reconstructs $\hat{x} = D(\hat{\mathbf{z}})$. The methods differ almost entirely in Q; the encoder, decoder, and reconstruction objective are shared.

Hard assignment and collapse. The canonical quantizer [1] assigns each token to its nearest code, $k^\star = \arg\min_k \|\mathbf{z} - \mathbf{e}_k\|_2^2$. Since arg min is non-differentiable, it is trained with a straight-through estimator (STE) [8] that copies the decoder gradient onto the encoder output, plus a *commitment* term $\beta \|\mathbf{z} - \text{sg}[\mathbf{e}_{k^\star}]\|_2^2$ ($\text{sg}[\cdot]$ is stop-gradient) which keeps encoder outputs near their assigned codes. Because assignment is hard, only the selected code per token receives a gradient; rarely-chosen codes drift and stop being selected, causing *codebook collapse*, where many codes go unused (*dead codes*) and the effective vocabulary is far smaller than K. Standard fixes like EMA updates or dead-code resets treat the symptom rather than the assignment rule. We measure this through *utilization*, the fraction of codes used at least once over an evaluation set; keeping it high without sacrificing the adaptivity of a learnable codebook is the challenge the methods in Sec. 5 address, and the axis along which we compare them with LGQ. Soft-assignment methods, the family LGQ belongs to, route gradient to all K codes at once (Sec. 3).

3 Method: Learnable Geometric Quantization

3.1 Setup

We instantiate the tokenizer of Sec. 2 (Fig. 2) with $T = 256$ spatial tokens (for 256×256 images at the downsampling factor $f = 16$) and the embedding dimension $C = 64$. The quantizer maps each token $\mathbf{z}_{b,t} \in \mathbb{R}^C$ to a discrete code $k \in \{1, \dots, K\}$ and embedding $\hat{\mathbf{z}}_{b,t}$.

3.2 Soft-assignment with Learnable Codes

LGQ maintains K learnable codes $\mathbf{e}_k \in \mathbb{R}^C$ and computes a soft-assignment using squared distance ℓ_2 and temperature τ:

$$p_{b,t,k} = \frac{\exp\left(-\|\mathbf{z}_{b,t} - \mathbf{e}_k\|_2^2/\tau\right)}{\sum_{k'} \exp\left(-\|\mathbf{z}_{b,t} - \mathbf{e}_{k'}\|_2^2/\tau\right)}. \tag{1}$$

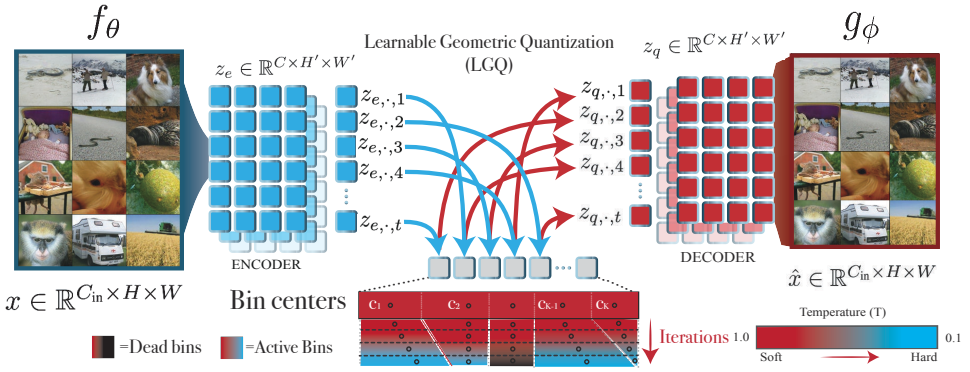


Figure 2 | LGQ tokenizer. The encoder f_θ maps an image to continuous latents z_e; LGQ assigns each token to learnable codes via the annealed soft-assignment ($\tau : 1.0 \to 0.1$), and the decoder g_ϕ reconstructs from the quantized z_q. The same backbone is shared by all six quantizers in our comparison.

The decoded vector uses straight-through gradients [8]:

$$\hat{\mathbf{z}}_{b,t} \;=\; \mathbf{e}_{k^*} \;+\; \underbrace{\sum_k p_{b,t,k}\mathbf{e}_k \;-\; \mathrm{sg}\left[\sum_k p_{b,t,k}\mathbf{e}_k\right]}_{\text{soft path for gradient}}, \tag{2}$$

where $k^* = \arg\max_k p_{b,t,k}$ and $\mathrm{sg}[\cdot]$ is the stop-gradient. The forward pass therefore commits to a single discrete code (as in standard VQ), while the backward pass flows through the full soft-assignment, giving the codebook geometry a well-defined gradient signal at every step. The score is the squared distance $\|\mathbf{z} - \mathbf{e}_k\|_2^2$ rather than the unnormalized inner product $\langle \mathbf{z}, \mathbf{e}_k \rangle$ used by IBQ [9]; Sec. 5.2 sets the two rules against each other. Squared ℓ_2 is also numerically convenient under the annealing schedule below: at small τ the exponent saturates the softmax to a near-hard one-hot without a separate safeguard.

3.3 Temperature Annealing

We linearly anneal τ from $\tau_{\text{start}} = 1.0$ to $\tau_{\text{end}} = 0.1$ over training. Early high temperature gives near-uniform assignments and an exploration phase in which codes can move freely and respond to the encoder distribution; late low temperature commits each token to one code and lets the codebook geometry crystallize. We ablate both endpoints and the schedule shape in Sec. A.2.

3.4 Regularization

Let $\bar{p}_k = \frac{1}{BT}\sum_{b,t} p_{b,t,k}$ be the average usage of code k over the $B{\cdot}T$ tokens of one batch. Under data-parallel training $\bar{p}$ is formed independently on each device from that device's local batch, with no cross-rank reduction, so B denotes the per-GPU batch size throughout. We add two terms to the reconstruction loss:

$$\mathcal{L}_{\text{peak}} = \frac{1}{BT}\sum_{b,t} \mathrm{ReLU}\left(1 - \sum_k p_{b,t,k}^2\right), \tag{3}$$

$$\mathcal{L}_{\text{div}} = K \sum_k \bar{p}_k^2. \tag{4}$$

$\mathcal{L}_{\text{peak}}$ encourages each token's soft-assignment distribution to concentrate on a single code so that the soft-assignment behaves like a near-hard one-hot at training time (*peakedness*); $\mathcal{L}_{\text{div}}$ penalizes concentration of average usage in the batch. The two terms are deliberately push-pull: peakedness asks individual tokens to commit to a single code, while diversity asks the batch as a whole to spread its commitments across the codebook. The K-scaled diversity term is the load-bearing anti-collapse pressure while peakedness is a milder refinement that improves per-token commitment for a further utilization and rFID gain. The regularizer-weight ablation in Sec. A.1 isolates both effects.

Why $\mathcal{L}_{\text{div}}$ carries a factor of K. The K scaling normalizes the term against the uniform usage vector: if all $\bar{p}_k = 1/K$ then $K \sum_k \bar{p}_k^2 = 1$ for any K. This normalization is exact while assignments remain soft and $\bar{p}$ retains full support. Once $\mathcal{L}_{\text{peak}}$ and the annealed temperature drive assignments toward one-hot, $\bar{p}$ is an average of $B{\cdot}T$ indicators and cannot spread over more than $B{\cdot}T$ codes, so the smallest value the term can take grows with K; Sec. 4 states the bound in both regimes. We use one λ_{div} across $K = 4{,}096$ to $65{,}536$, and Table 3 reports the sweep it produces, where $\lambda_{\text{peak}} = \lambda_{\text{div}} = 0.005$.

Total loss.

$$\mathcal{L} = \mathcal{L}_{\text{recon}} + \lambda_{\text{peak}}\mathcal{L}_{\text{peak}} + \lambda_{\text{div}}\mathcal{L}_{\text{div}}. \tag{5}$$

$\mathcal{L}_{\text{recon}}$ is the VQ-GAN reconstruction objective [10]: a mixture of a pixel-space ℓ_1 term, an LPIPS perceptual term [11], and a PatchGAN hinge loss,

$$\mathcal{L}_{\text{recon}} = \lambda_{\ell_1}\|x - \hat{x}\|_1 + \lambda_{\text{lpips}}\,\mathcal{L}_{\text{LPIPS}}(x, \hat{x}) + \lambda_{\text{gan}}\,\mathcal{L}_{\text{GAN}}(\hat{x}), \tag{6}$$

where x is the input image, $\hat{x}$ the decoded reconstruction, and $\mathcal{L}_{\text{GAN}}(\hat{x}) = -\mathbb{E}[D(\hat{x})]$ is the generator term of the PatchGAN hinge loss with discriminator D trained via $\mathbb{E}[\max(0, 1 - D(x))] + \mathbb{E}[\max(0, 1 + D(\hat{x}))]$. We set λ_{ℓ_1}=1.0, λ_{lpips}=1.0, and λ_{gan}=0.1; the discriminator is enabled after 10,000 optimizer steps. All six quantizers in our comparison are trained with this identical reconstruction objective. The only change between runs is the quantizer.

4 Theoretical Properties

Although LGQ is presented as an empirical method, several properties of its assignment rule admit clean proofs that motivate the regularizer and schedule choices in Sec. 3.

4.1 Preliminaries: Annealed-Softmax Properties

LGQ's soft-assignment is the temperature-scaled softmax $p_k^\star = \exp(-\|\mathbf{z} - \mathbf{e}_k\|_2^2/\tau)/Z$ (Eq. (1)), and inherits three standard (non-LGQ-specific) properties of the Gibbs distribution that we use below. *(i)* It is the unique simplex minimizer of the free energy $\sum_k p_k\|\mathbf{z} - \mathbf{e}_k\|_2^2 + \tau \sum_k p_k \log p_k$, so τ trades distortion against entropy. *(ii)* As $\tau \to 0^+$ it converges to the hard nearest-neighbour indicator $\mathbb{I}[k{=}k^\star]$, so annealing $\tau : 1.0 \to 0.1$ interpolates from an exploratory near-uniform prior to committed quantization. *(iii)* For $\|\mathbf{z}\|, \|\mathbf{e}_k\| \le M$ the assignment Jacobian satisfies $\sum_k \|\nabla_{\mathbf{z}} p_k\|_2 \le 4M/\tau$, so the soft path is Lipschitz and the STE backward signal stays bounded across the schedule.

4.2 Bridging VQ and FSQ

By Sec. 4.1, LGQ's assignment recovers VQ's hard nearest-neighbour rule as $\tau \to 0$ yet stays smooth with a bounded Jacobian for any $\tau > 0$, so the codebook receives a non-degenerate gradient on all K codes at every step where VQ's arg min routes gradient through exactly one code per token. The K-scaled diversity term pushes toward FSQ-style full utilization without freezing the codebook to a grid. LGQ inherits VQ's adaptivity and FSQ's collapse-resistance, with the regularizer pair acting as the bridge between the two regimes.

4.3 Bounds on the Regularizers

Proposition 1 | Peakedness lower bound. *For any $p \in \Delta^{K-1}$, $\sum_k p_k^2 \le 1$ with equality iff p is one-hot. Hence $\mathcal{L}_{\text{peak}} = \mathbb{E}_n[\text{ReLU}(1 - \sum_k p_{n,k}^2)] \ge 0$, with equality iff every token's assignment is one-hot.*

Proposition 2 | Lower bound on the diversity term. *Let $S = \{k : \bar{p}_k > 0\}$ be the support of $\bar{p}$ and let $m = |S|$ be its size. Then*

$$\mathcal{L}_{\text{div}} = K \sum_k \bar{p}_k^2 \ge \frac{K}{m},$$

with equality iff $\bar{p}$ is uniform on S. If every token's assignment is one-hot, $\bar{p}$ is an average of $B{\cdot}T$ indicator vectors, so $m \le \min(B{\cdot}T, K)$ and $\mathcal{L}_{\text{div}} \ge K / \min(B{\cdot}T, K)$.

Both propositions follow from Cauchy–Schwarz; the proofs are given in Sec. B.

Which value of m applies depends on where training is. Early on the temperature is high, each token's assignment is close to uniform, $\bar{p}$ has full support, and the bound reads $\mathcal{L}_{\text{div}} \ge 1$ for every K. Late in training $\mathcal{L}_{\text{peak}}$ and the annealed temperature commit each token to a single code, and the bound becomes $K/(B{\cdot}T)$ once $B{\cdot}T$ is smaller than K, so the floor the diversity term is pushed against grows with the codebook while λ_{div} is held fixed. Replacing the K prefactor with $\min(B{\cdot}T, K)$ restores a floor of 1 at every K; our runs use the K prefactor, and Table 3 reports what it delivers across the sweep.

The two bounds motivate a regularizer-weight ablation, reported in full in Sec. A and summarized here. These are reduced-budget runs (50,000 images, 10 epochs), so their absolute scores sit well below the full-budget results of Table 2 and should be read as relative effects. Ablating the terms individually at K=16,384: dropping peakedness leaves the model usable (53.40 to 56.47 rFID, utilization 47.7 to 38.4%), whereas dropping diversity destroys it (53.40 to 204.55 rFID, utilization 47.7 to 0.12%, i.e., roughly 20 of 16,384 codes alive). Removing both is no worse on rFID than removing diversity alone (174.19 against 204.55) and equally collapsed (0.07% against 0.12% utilization). Diversity is therefore the load-bearing anti-collapse pressure and peakedness a refinement on top of it, an ordering rather than a pair of equals.

5 Related Work

5.1 Baseline Quantizers and Their Failure Modes

We organize the related work around two questions: *how does each method prevent codebook collapse, and at what design cost?* Sec. D tabulates the answers method by method.

Each baseline answers the collapse question by restricting something: where codes may sit, how many of them receive gradient, or what the assignment is scored by. VQ [1] routes gradient through the arg min to a single code per token, so unselected codes drift into dead codes; EMA updates and resets treat the symptom. RotVQ [7] repairs the gradient path with a rotation and rescaling but leaves the hard assignment in place. FSQ [6] and LFQ [4] remove collapse by construction, FSQ by fixing an axis-aligned scalar grid and LFQ by restricting codes to the corners of a sign-pattern hypercube, and both give up the ability to place capacity where the data is dense. SimVQ [5] and VQGAN-LC [12] keep a codebook but tie it to a shared learned transform or to frozen pretrained features, so individual codes cannot migrate. IBQ [9] is closest to LGQ: it applies STE to a softmax over all K codes, so every code receives a gradient, but scores them by the unnormalized dot product $\mathbf{z}^\top \mathbf{e}_k$ under a fixed temperature with an entropy penalty on the assignment distribution, where LGQ uses squared ℓ_2, an annealed τ, and regularizers on the soft assignment. Per-method detail is given in Sec. D.

Positioning. LGQ is subject to none of the three restrictions above. Codes move independently, all codes receive gradient, and the score is the squared Euclidean distance, the same quantity the reconstruction loss minimizes. Utilization is controlled by the regularizers of Sec. 4 rather than by the codebook parameterization, and one $(\lambda_{\text{peak}}, \lambda_{\text{div}})$ holds from K=4,096 to K=65,536 without per-K retuning, which a fixed or tied codebook does not require and a hard-assignment learned codebook does not achieve.

5.2 ℓ_2^2 and Dot-Product Logits Comparison (LGQ vs. IBQ)

The closest baseline to LGQ at the algorithmic level is IBQ [9], which shares our two central design choices: it applies STE to a softmax over all K codes, so every code receives a gradient at every step rather than only the selected one. The methods diverge in a single but consequential place, which is the quantity the softmax scores. IBQ scores codes by the unnormalized inner product $\langle \mathbf{z}, \mathbf{e}_k \rangle$, whereas LGQ scores them by the squared ℓ_2 distance $-\|\mathbf{z} - \mathbf{e}_k\|^2$.

The distinction is clearest from the expansion of the squared distance:

$$\|\mathbf{z} - \mathbf{e}_k\|^2 = \|\mathbf{z}\|^2 \; - \; 2\langle \mathbf{z}, \mathbf{e}_k \rangle \; + \; \|\mathbf{e}_k\|^2. \tag{7}$$

The MSE reconstruction loss therefore decomposes into three terms: the encoder-output norm $\|\mathbf{z}\|^2$, the code norm $\|\mathbf{e}_k\|^2$, and the inner product $\langle \mathbf{z}, \mathbf{e}_k \rangle$ that couples them. For a fixed token, the first term $\|\mathbf{z}\|^2$ is a constant shared by every code, so it has no effect on which code minimizes the distance; the assignment is decided entirely by the remaining two terms, $-2\langle \mathbf{z}, \mathbf{e}_k \rangle + \|\mathbf{e}_k\|^2$. LGQ's distance-based score retains both, which is exactly the Bayes-optimal nearest-neighbor quantizer for the MSE objective. IBQ's dot-product score keeps only the inner-product term and *completely eliminates* $\|\mathbf{e}_k\|^2$.

The consequence is a systematic bias. Because IBQ's loss contains no term that constrains or compensates for the code norms, a code can win an assignment by sheer magnitude rather than by its proximity to $\mathbf{z}$. Enlarging $\|\mathbf{e}_k\|$ inflates $\langle \mathbf{z}, \mathbf{e}_k \rangle$ and makes that code more likely to be selected, even when a smaller-norm code lies closer in Euclidean terms. The dot-product rule thus coincides with the optimal rule only in the degenerate case where all $\|\mathbf{e}_k\|$ are equal. Once the norms vary, it systematically prefers larger-norm codes, with per-sample excess distortion bounded by the

Table 1 | Matched logit ablation. Identical backbone, seed, schedule, and compute; only the quantity the softmax scores differs. rFID is the mean over a matched three-epoch window, since a single epoch is within run-to-run noise. "Active" is the number of codes assigned at least once on the validation set, out of K=16,384.

Softmax score	rFID ↓	Util. % ↑	Active
$-\|\mathbf{z}-\mathbf{e}_k\|^2$ (LGQ)	**12.26**	**100.00**	**16,384**
$\langle \mathbf{z}, \mathbf{e}_k \rangle$ (IBQ-style)	13.62	79.86	13,084

maximum norm gap $\max_{j,k}(\|\mathbf{e}_j\|^2 - \|\mathbf{e}_k\|^2)$. This is a structural property of the selection rule, and the bias grows with the heterogeneity of the learned codebook.

We treat the realized effect empirically rather than as a proof. The IBQ–LGQ rFID gap (16.21 vs. 12.32 at K=16,384) motivates the ℓ_2^2 choice but does not by itself isolate it, since published IBQ also differs in its two-sided (double) quantization loss, backbone depth, and entropy penalty.

Isolating the logit. We rerun LGQ changing *only* the quantity the softmax scores, holding backbone, seed, schedule, optimizer and budget fixed (Table 1). Reconstruction separates the two arms by 1.36 rFID over a matched three-epoch window (12.26 against 13.62), against an epoch-to-epoch scatter of 0.92 rFID between them. Code recruitment separates them by much more. The dot-product arm has assigned the same 13,084 codes throughout, with 3,300 codes never assigned once and none recovered, while the ℓ_2^2 arm climbs to 100% of the codebook and holds it. This is the norm term of Eq. (7) made visible. The dot-product score is unbounded above in $\|\mathbf{e}_k\|$, so a code raises its score by growing rather than by moving closer to the data. A large code wins assignments it is not nearest to, and a code that stops being selected receives no assignment gradient with which to grow back. The squared distance admits no such shortcut, since its score is maximized only by a code sitting exactly on the data point. The codebook norms confirm this. At the last audited checkpoint the 3,300 dead codes have $\|\mathbf{e}_k\| \in [0.51, 0.63]$ against $[5.20, 5.85]$ for the 13,084 live ones, two disjoint ranges with $\mathrm{corr}(\|\mathbf{e}_k\|, \log \bar{p}_k) = 0.997$, and every dead norm shrinks between consecutive checkpoints. Under $-\|\mathbf{z}-\mathbf{e}_k\|^2$ the same correlation is 0.08 and no code is dead (Sec. H). The ablation isolates the score with LGQ's regularizers held fixed. Published IBQ pairs the same dot-product score with a low-temperature entropy penalty on the assignment distribution and reaches 99.99% utilization (Table 2). That containment is size-specific: at K=65,536 under the identical protocol, IBQ's utilization peaks at 98.2% and falls to 55.1% while LGQ's rises to 100% (Sec. H). The misallocation the score permits re-emerges as the codebook grows (Sec. 7.2).

The two propositions and their proofs, together with a probabilistic (Gaussian mixture vs. von Mises–Fisher) reading of the two selection rules and the gradient/compute trade-off they imply, are given in Sec. H.

Relation to other codebook designs and regularizers. Residual and multi-stage quantizers such as RQ-VAE [13] address large-K capacity by composing several small codebooks; LGQ is orthogonal to this axis as a single-stage learnable codebook, and the two are combinable. Nor are our regularizers new in isolation: usage and entropy penalties, temperature-annealed soft-assignment, and geometric spreading terms all appear in prior VQ variants (Sec. D). The contribution is the specific combination, ℓ_2^2 soft-assignment with annealing and the push-pull peakedness and K-scaled

diversity pair, which removes collapse without EMA, resets, or reparameterization and holds across the complete K sweep under one setting.

6 Experimental Setup

6.1 Backbone

All quantizers share an identical VQ-GAN-style CNN autoencoder backbone [10] with base channel width 128 (widening to 256 at the deepest stage) and embedding dimension $C = 64$, downsampling factor $f = 16$, producing $T = 256$ spatial tokens for 256×256 images. The encoder/decoder include residual blocks and self-attention at the 16×16 feature map. The replacement of the quantizer is the only change between runs in Table 2. The complete pipeline is diagrammed in Fig. 2. Training uses DDP across 4 GPUs at per-GPU batch size 8, so each rank holds $B{\cdot}T{=}2048$ tokens per batch; the full optimizer and data settings are in Sec. E.

6.2 Evaluation Protocol

Reconstruction. We report PSNR, SSIM [14], LPIPS [11], reconstruction loss and codebook utilization (fraction of entries used at least once on the validation set). We also report **rFID** [15]: Fréchet Inception Distance between encoder–decoder reconstructions of the validation set and a matched real-image reference set, computed against a frozen reference. Every rFID we report is computed on a fixed 10,000-image subset of the ImageNet validation split, identical for every row, so the values are comparable within this paper but are not comparable to published rFID computed on all 50,000 images. Recomputing on the full split at each run's best checkpoint leaves the ordering unchanged (LGQ 6.05, SimVQ 8.19, IBQ 8.42, FSQ 8.83, RotVQ 12.76).

Training runs and seeds. Each tokenizer is trained once per setting: every row of Tables 1 to 5 comes from a single training run. The ± values in Table 5 are computed over four sampling seeds at a fixed trained tokenizer and quantify sampling variability alone. Per-epoch reconstruction curves for every method, from which the epoch-to-epoch scatter of a single run can be read directly, are given in Sec. I.

Best-epoch reporting under a matched compute budget. All methods are trained under an identical wall-time budget, with periodic checkpointing and the same validation cadence. For each method, we report the best-epoch checkpoint reached within the budget, the standard early-stopping protocol, so that within-paper comparisons reflect each method's best quality under matched compute rather than an arbitrary cutoff.

6.3 Downstream MaskGIT Stage

The choice of MaskGIT, rather than the larger LlamaGen-style autoregressive transformers used by IBQ [9] or the LM-style decoder of MAGVIT-v2 [4], is deliberate: at our scale it isolates the tokenizer's contribution without conflating it with a multi-billion-parameter decoder. After tokenizer training, each tokenizer is frozen and used to convert images to discrete token sequences. On top of the frozen tokenizer we train a 50.8M-parameter MaskGIT-style bidirectional transformer [3] (12 layers, hidden size 512) to predict randomly masked visual tokens, using a cosine masking schedule, a start-of-sequence token, and a separate mask-token ID. We train on the full ImageNet training

Table 2 | Reconstruction at $K = 16{,}384$, ImageNet 256×256, identical backbone. Best-epoch validation metrics from each run is reported. All six methods are trained under the same matched compute budget; we report the lowest-rFID checkpoint reached for each. **Bold** marks the best in each metric column.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	Rec. loss ↓	rFID ↓	Util. %	Perplexity
RotVQ	20.572	0.5788	0.3032	0.2635	20.15	89.83	1,921
LFQ	21.124	0.6013	0.2858	0.2450	18.82	100.00	16,046
FSQ	21.475	0.6014	0.2705	0.2322	17.38	100.00	13,651
IBQ	21.339	0.6145	0.2661	0.2373	16.21	99.99	15,101
SimVQ	20.930	0.5990	0.2778	0.2512	16.05	100.00	3,981
LGQ (ours)	**21.791**	**0.6289**	**0.2438**	**0.2225**	**12.32**	100.00	**16,011**

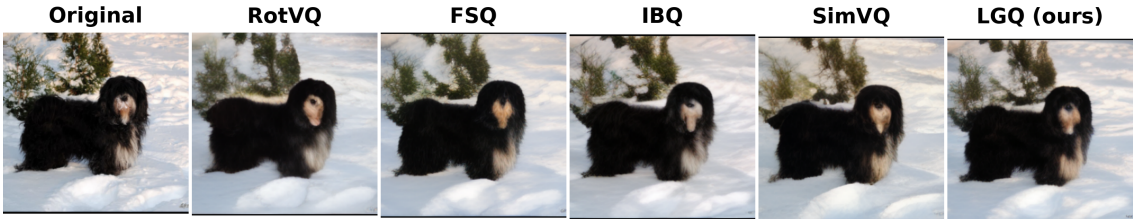


Figure 3 | Reconstructions at K=16,384. One ImageNet validation image, then its reconstruction from five of the six frozen tokenizers. LGQ retains the finest texture and color fidelity, consistent with its best rFID/LPIPS in Table 2. The full per-method grid over six images is in Sec. K.

set (1,281,167 images, 1,000 classes) at 256×256 with codebook size $K = 16{,}384$, an effective batch of 128 (per-step batch 32 with gradient accumulation 4), AdamW at lr $= 10^{-4}$, and early stopping on the validation NLL with patience 8–10. All tokenizers are trained under matched settings, and the leading tokenizers reach ∼100 epochs, by which point validation NLL has plateaued.

For the generation-side comparison (Sec. 7.4, Table 5) we follow MAGVIT-v2 [4] and IBQ [9] in reporting gFID, and Inception Score where available, on 50K samples drawn from the trained sampler under the standard cosine schedule with 11 decoding steps.

7 Results

7.1 Reconstruction at $K = 16{,}384$

Table 2 reports the validation-set reconstruction metrics rFID, PSNR, SSIM, LPIPS, reconstruction loss, codebook utilization, and perplexity for each method. All six runs share the identical ℓ_1 + LPIPS + PatchGAN reconstruction objective described in Sec. 3. The only change between runs is the quantizer. LGQ achieves the best PSNR, SSIM, LPIPS, and rFID across the group; Fig. 3 shows the corresponding reconstructions. Table 2 carries no plain-VQ row: RotVQ serves as the learnable-argmin baseline, and VQ is retained in Sec. D as a design reference.

7.2 LGQ Codebook-Size Sweep

Table 3 reports LGQ across five codebook sizes. PSNR, SSIM, and LPIPS improve with K, and rFID drops monotonically from 16.91 at $K = 4{,}096$ to 10.72 at $K = 65{,}536$. The key result is that

Table 3 | LGQ codebook-size sweep on ImageNet 256×256**.** For each K we report the converged checkpoint at which codebook utilization saturates to 100%. **Bold** marks the best in each metric column. A single set of regularizer weights (λ_{peak}=λ_{div}=0.005) and a single temperature schedule (1.0 → 0.1 linear) is used across the entire sweep without per-K retuning.

K	PSNR ↑	SSIM ↑	LPIPS ↓	Rec. loss ↓	rFID ↓	Util. %	Perplexity
4,096	21.353	0.6080	0.2751	0.2358	16.91	100.00	3,997
8,192	21.494	0.6167	0.2610	0.2312	14.59	100.00	7,893
16,384	21.791	0.6289	0.2438	0.2225	12.32	100.00	16,011
32,768	21.932	0.6386	0.2371	0.2180	11.90	100.00	31,826
65,536	**22.183**	**0.6491**	**0.2307**	**0.2123**	**10.72**	100.00	**63,077**

utilization reaches and holds 100% at every K, so the learnable codebook never collapses as it grows 16×, under a single fixed set of regularizer weights with no per-K retuning. What changes with K is how long saturation takes: the first epoch at which utilization reaches 100% is epoch 7 at K=4,096, then 10, 13, 22 and 29 as K grows to 65,536, and once reached it is held at every later epoch we measure. Each row is read at or after its own saturation epoch, so a larger codebook is given the extra epochs it needs rather than a checkpoint chosen to flatter it. All four reconstruction metrics continue to improve through K = 65,536 with no sign of levelling off at this backbone capacity. To test whether this behaviour is specific to LGQ, Sec. H trains IBQ at K=4,096, 16,384 and 65,536 under the identical protocol: LGQ is ahead on every metric at all three sizes at a matched epoch, and where IBQ's utilization falls as K grows, LGQ's rises to 100% (Sec. 5.2).

7.3 Codebook Geometry Analysis

To move beyond aggregate rFID and characterize *how* each codebook distributes information, we encode 5,000 ImageNet validation images through each tokenizer and compute four statistics over the resulting (N, T) index tensors (T=256 spatial tokens per image, K=16,384): normalized **marginal entropy** H_{norm}=$H(p)/\log K$ (usage uniformity across tokens: 1 is perfectly uniform, low values flag dominant "magnet codes"); **top-1% mass** (the share of probability captured by the most-used 1% of codes, low is balanced); **co-occurrence density** (the fraction of code-pairs that ever appear within the same image; high indicates compositional, freely-combining codes); and **pairwise mutual information** between two random spatial positions (high indicates structured, non-redundant spatial coding).

Table 4 shows that LGQ's codebook is near-perfectly uniform (H_{norm}=0.997), so no small subset of codes absorbs the token distribution. Its top-1% mass of only 1.9% confirms that no "dead star" codes dominate; in contrast, RotVQ allocates 43% of total usage to its top 164 codes: a ∼23× concentration relative to LGQ.

The co-occurrence density (13.1%) and pairwise MI (8.08 nats) are also highest for LGQ, indicating codes that combine richly across spatial positions; IBQ is the closest competitor (co-occurrence 12.9%, MI 8.01), with FSQ just behind. While FSQ and SimVQ also achieve 100% utilization, their lower mutual information and co-occurrence scores indicate that global utilization alone does not guarantee a rich latent space. A UMAP overlay of the learned codes on the continuous encoder distribution, given in Sec. J, shows the same thing: the baselines group into isolated subregions

Table 4 | Codebook geometry at K=16,384. LGQ leads on all four metrics; IBQ is second on each. RotVQ is severely concentrated, with 43% of usage in its top 1% of codes. Utilization here is on the 5,000-image analysis subset, so RotVQ reads lower than its full-validation value in Table 2. MI is a plug-in estimate over 32 random spatial position pairs from the same 5,000 images. That estimator saturates near $\log N$=8.52 nats, close to the values reported here, so the MI column should be read as a ranking and not as a measurement of absolute mutual information.

Tokenizer	Util. %	H_{norm} ↑	Top-1% ↓	Co-occ. % ↑	MI ↑
LGQ (ours)	**100.0**	**0.997**	**1.9**	**13.1**	**8.08**
IBQ	100.0	0.991	2.6	12.9	8.01
FSQ	100.0	0.977	3.7	12.2	7.94
SimVQ	100.0	0.854	28.9	6.2	6.34
RotVQ	79.9	0.771	43.4	4.0	5.20

where LGQ spreads smoothly across the distribution.

Codebook uniformity is positively but imperfectly associated with reconstruction quality (Table 2): the ordering is anchored by LGQ at both extremes but is not monotone, SimVQ reaches a strong rFID (16.05) at only moderate uniformity (H_{norm}=0.854), so uniformity is best read as one contributing factor, which LGQ's regularizers pursue as a mechanism to keep the full codebook available to the decoder rather than as an end in itself.

7.4 Downstream Generation with MaskGIT

Using the MaskGIT transformers trained on each frozen tokenizer, we report the class-conditional 50K-sample generation comparison at K = 16,384 in Table 5; per-tokenizer sample grids are in Sec. K. LGQ attains the best gFID (57.69), extending its reconstruction lead into generation, with IBQ second (59.47); IBQ attains the best IS (15.03). Ordered by gFID, LGQ produces the sharpest and most class-coherent samples and RotVQ (92.99 gFID) the blurriest.

Utilization is not sufficient. FSQ, SimVQ, and LGQ all reach 100.00% utilization at K=16,384 (Table 2) yet span 12.5 gFID (70.16, 69.38, 57.69), so a fully used codebook does not by itself determine generation quality: high utilization shows that codes are assigned, not that quantization error is low or that the resulting token distribution is easy to model. Our claim is that LGQ attains high utilization, the best reconstruction, and the best generation *jointly*. The finer usage statistics of Sec. 7.3 do separate the fully-utilizing methods and order them consistently with quality, but they too are measured alongside quality rather than intervened upon; a causal test would require intervening on the token distribution with reconstruction held fixed, which we do not attempt.

7.5 Computational Cost

All-code soft assignment is widely assumed to be expensive; the full timing and memory table is in Sec. C. Timed in isolation at the training configuration, the quantizer accounts for ~10% of a full optimizer step at K=16,384, where LGQ is 3.8× faster and 1.5× smaller than IBQ; IBQ exhausts memory at K=262,144, where LGQ still trains. FSQ is cheaper than both and K-independent, so our claim is best reconstruction per unit cost, not lowest cost. End-to-end the picture is flatter

Table 5 | Class-conditional MaskGIT generation at K=16,384. gFID and IS are measured on 50K class-conditionally sampled images (T=11 steps, identical sampling settings across tokenizers) against ImageNet-256 reference statistics, following MAGVIT-v2 [4] and IBQ [9]. Each tokenizer is a single training run; for LGQ, IBQ, and FSQ we report mean ± std over four *sampling* seeds at that fixed tokenizer. Each MaskGIT is trained on its method's *best-rFID* tokenizer and evaluated at its best-val-NLL checkpoint within a matched wall-clock budget.

Tokenizer	rFID ↓	gFID ↓	IS ↑
LGQ (ours)	**12.32**	**57.69 ±0.26**	14.80
IBQ	16.21	59.47 ±0.09	**15.03**
SimVQ	16.05	69.38	13.26
FSQ	17.38	70.16 ±0.24	13.60
RotVQ	20.15	92.99	10.63

still: one epoch costs 15–17 GPU-hours for *every* quantizer at K=16,384, so the quantizer is not the end-to-end bottleneck at any K we train at.

8 Discussion

Under a fixed VQ-GAN-style objective with all quantizers sharing an identical backbone and budget, LGQ achieves the best rFID, a −3.73 gap to the next baseline, from a single untuned configuration; in the ablations of Sec. A rFID stays within a narrow band along the symmetric diagonal from λ_{peak}=λ_{div}=0.001 to 0.05 and across annealing-schedule shapes. The advantage holds as the codebook grows 16× to K=65,536 with utilization saturated throughout, and extends to downstream generation, where LGQ attains the best gFID. Notably, the baseline ordering does not transfer cleanly from reconstruction to generation, echoing prior reports that stronger reconstruction need not yield better generation.

Limitations and future work. This study has three principal limitations. First, our evaluation is confined to a single modality: all experiments are conducted on static images at ImageNet 256×256, and we do not assess whether LGQ's advantages extend to other modalities such as audio or video. Second, the codebook-size sweep covers LGQ at five values of K and one baseline, IBQ, at K=4,096, 16,384 and 65,536; the remaining quantizers are evaluated only at the matched size K=16,384. We show that LGQ's K-scaled diversity regularizer stays calibrated across the sweep without re-tuning and that IBQ loses utilization over the same range, but we do not characterize how the remaining quantizers scale; extending the sweep to every quantizer is left to future work. Third, each tokenizer is trained once per setting, so we report no variance across training seeds; the intervals in Table 5 are over sampling seeds at a fixed tokenizer, and the per-epoch curves in Sec. I are the only evidence we offer on run-level stability. We further note that, because our shared backbone and matched training budget differ from those of the original baseline papers, we regard the *relative* ranking obtained under this identical-backbone protocol as the claim of this work, rather than the absolute scores.

9 Conclusion

We introduced Learnable Geometric Quantization (LGQ), a vector quantizer that eliminates codebook collapse without EMA or reset heuristics. By combining soft-to-hard temperature annealing with lightweight peakedness and K-scaled diversity regularizers, LGQ scales to large codebooks while maintaining 100% utilization under one hyperparameter setting. On ImageNet 256×256 within a fixed VQ-GAN backbone, it achieves the best rFID, PSNR, SSIM, and LPIPS at K=16,384, and keeps improving as the codebook grows to K=65,536. Trained on the frozen tokenizers, MaskGIT transformers achieve the best class-conditional gFID with LGQ, leading on reconstruction and generation alike.

DATA AND CODE AVAILABILITY

Code and training configurations are available at https://anonymous.4open.science/r/lgq-anon-E12C/. ImageNet ILSVRC-2012 [16] is public and cited in Sec. 6.

COMPETING INTERESTS

The authors declare no competing interests.

References

[1] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

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

[3] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. MaskGIT: Masked generative image transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[4] Lijun Yu, José Lezama, Nitesh B Gundavarapu, Luca Versari, Jascha Sohl-Dickstein, Matthias Minderer, Leonidas Guibas, Uri Alon, Vighnesh Birodkar, Vittorio Fischetti, et al. Language model beats diffusion – tokenizer is key to visual generation. In *International Conference on Learning Representations (ICLR)*, 2024.

[5] Yongxin Zhu, Bocheng Li, Yifei Xin, Zhihua Xia, and Linli Xu. Addressing representation collapse in vector quantized models with one linear layer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025.

[6] Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: VQ-VAE made simple. In *International Conference on Learning Representations (ICLR)*, 2024.

[7] Christopher Fifty, Ronald Junkins, Dennis Duan, Aniketh Iyengar, Jerry Liu, Ehsan Amid, Sebastian Thrun, and Christopher Ré. Restructuring vector quantization with the rotation trick. In *International Conference on Learning Representations*, volume 2025, pages 19153–19188, 2025.

[8] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[9] Fengyuan Shi, Zhuoyan Luo, Yixiao Ge, Yujiu Yang, Ying Shan, and Limin Wang. Scalable image tokenization with index backpropagation quantization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16037–16046, 2025.

[10] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[11] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[12] Lei Zhu, Fangyun Wei, Yanye Lu, and Dong Chen. Scaling the codebook size of VQ-GAN to 100,000 with a utilization rate of 99%. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[13] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[14] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[15] 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 (NeurIPS)*, 2017.

[16] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[17] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

[18] Stuart P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2): 129–137, 1982.

APPENDIX

A Ablations

We isolate the two load-bearing design choices in LGQ: the regularizer weights (λ_{peak}, λ_{div}) and the temperature schedule (τ_{start}, τ_{end}, schedule shape). All ablation runs use the same backbone and reconstruction objective as the main comparison (Sec. 6), $K = 16{,}384$, trained on a reduced 50,000-image / 10-epoch budget so the full grid is tractable. Absolute scores are therefore lower than the full-budget results of Sec. 7 and should be read as *relative* effects; row 1 of each table is the default setting at this same reduced budget.

A.1 Regularizer-Weight Ablation

Table A1 sweeps λ_{peak} and λ_{div} jointly around the default 0.005 setting, and also disables each regularizer in isolation. The two "λ=0" rows test the necessity of each term: the push-pull formulation of Sec. 3.4 predicts that peakedness alone collapses (peaked tokens, but only a few codes used across the batch) and that diversity alone leaves soft assignments diffuse (uniform usage, but no commitment to a single code at the token level). The 2D grid around 0.005 tests whether the chosen weight sits in a wide flat region or on a knife edge. Both predictions are borne out. Removing the diversity term (λ_{div}=0) collapses the codebook to 0.12% utilization (rFID 204.55), and disabling both terms is no better: it is equally collapsed (0.07% utilization) at a marginally lower rFID (174.19); even merely weakening diversity to 0.001 while keeping full peakedness already drops usage to 5.32%. Dropping peakedness alone (λ_{peak}=0) is far less damaging: utilization stays at 38.35% and rFID

Table A1 | Regularizer-weight ablation. LGQ at $K = 16{,}384$, varying λ_{peak} and λ_{div} jointly. Row 1 is the default setting; the "λ_{peak}=0" and "λ_{div}=0" rows test the necessity of each regularizer in isolation. Reduced-budget runs (see Sec. A); scores are relative, not comparable to the full-budget main results.

λ_{peak}	λ_{div}	PSNR ↑	SSIM ↑	LPIPS ↓	rFID ↓	Util. %	Notes
0.005	0.005	20.835	0.5833	0.3880	53.40	47.66	default
0	0.005	20.679	0.5626	0.3917	56.47	38.35	no peak: usable, weaker commitment
0.005	0	17.583	0.4369	0.5601	204.55	0.12	no div: codebook collapse
0	0	17.695	0.4390	0.5453	174.19	0.07	both off: collapse
0.001	0.001	20.796	0.5762	0.3718	50.48	21.61	weak regularization
0.001	0.005	20.586	0.5783	0.3788	49.74	40.30	asymmetric: weak peak, std div
0.005	0.001	20.304	0.5442	0.4224	75.59	5.32	asymmetric: std peak, weak div
0.01	0.01	20.947	0.5772	0.3636	51.29	48.85	2× default
0.05	0.05	20.787	0.5782	0.3796	52.08	68.24	10× default (over-reg. test)

degrades only mildly (53.40 → 56.47) which is consistent with the diversity term, not peakedness, being the load-bearing anti-collapse pressure. Across the symmetric grid from 0.001 to 0.05 rFID stays within a narrow 50.5–53.4 band, indicating the default 0.005 sits in a wide flat region rather than on an edge.

A.2 Temperature-Schedule Ablation

Table A2 ablates the temperature schedule. The default anneals τ linearly from 1.0 to 0.1 over training. We vary the starting temperature (0.5, 1.0, 2.0), the ending temperature (0.05, 0.1, 0.3), the constant-τ baseline (no annealing), and the schedule shape (linear vs. cosine). Two failure modes seemed likely a priori: starting or ending too hot should leave the soft assignment diffuse late in training, and starting or ending too cold should suppress early exploration, lowering utilization and raising rFID. The cold-start mode appears sharply. A constant cold schedule (τ=0.1) drops utilization to 4.33% at rFID 113.80, and a cooler start (0.5 → 0.1) already cuts usage to 8.37%. The hot-start mode, by contrast, does not appear at this budget: the hottest start (2.0 → 0.1) is the strongest row in the table (rFID 36.00, utilization 77.18%), ahead of the default 1.0 → 0.1 (53.40, 47.66%). For $\tau_{\text{start}} \geq 1.0$ the ending temperature, the cosine shape, and the constant-1.0 baseline all fall within noise of the default (52.8 to 54.1 rFID). The schedule is therefore forgiving on the warm side and brittle only when started cold. One consequence matters for how the main results should be read. The 1.0 → 0.1 schedule used for every main-table LGQ run is not the best setting in this ablation; a hotter start (2.0 → 0.1) is better here. We hold a single 1.0 → 0.1 schedule across the entire study rather than tuning per setting, so the headline LGQ numbers, which are already the best among all tokenizers, are reported at a conservative, untuned operating point. This ablation is run at reduced budget; whether the hot-start advantage persists at full budget is left to future work.

Table A2 | Temperature-schedule ablation. LGQ at $K = 16{,}384$, varying τ_{start}, τ_{end}, and the annealing shape. "Schedule" column: LIN = linear, COS = cosine, CONST = no annealing. Row 1 is the default setting. Reduced-budget runs (see Sec. A).

τ_{start}	τ_{end}	Schedule	PSNR ↑	SSIM ↑	LPIPS ↓	rFID ↓	Util. %
1.0	0.1	LIN	20.835	0.5833	0.3880	53.40	47.66
0.5	0.1	LIN	20.446	0.5548	0.4011	62.95	8.37
2.0	0.1	LIN	20.766	0.5807	0.3412	36.00	77.18
1.0	0.05	LIN	20.589	0.5622	0.3887	53.67	47.27
1.0	0.3	LIN	20.558	0.5804	0.3860	52.91	48.55
1.0	0.1	COS	20.680	0.5850	0.3863	54.06	50.56
1.0	1.0	CONST	20.638	0.5718	0.3820	52.78	52.86
0.1	0.1	CONST	19.378	0.4993	0.4852	113.80	4.33

B Proofs

This section proves the two propositions of Sec. 4.3, restated here so the appendix is self-contained.

Proposition | Restatement of Proposition 1. *For any $p \in \Delta^{K-1}$, $\sum_k p_k^2 \le 1$ with equality iff p is one-hot. Hence $\mathcal{L}_{\text{peak}} = \mathbb{E}_n[\text{ReLU}(1 - \sum_k p_{n,k}^2)] \ge 0$, with equality iff every token's assignment is one-hot.*

Proof of Proposition 1. By Cauchy–Schwarz applied to $p_k \cdot p_k$, $\sum_k p_k^2 \le (\max_k p_k)\sum_k p_k = \max_k p_k \le 1$, with both equalities iff some $p_{k^\star} = 1$. ∎

Proposition | Restatement of Proposition 2. *Let $S = \{k : \bar{p}_k > 0\}$ be the support of $\bar{p}$ and let $m = |S|$ be its size. Then $\mathcal{L}_{\text{div}} = K \sum_k \bar{p}_k^2 \ge K/m$, with equality iff $\bar{p}$ is uniform on S. If every token's assignment is one-hot, $\bar{p}$ is an average of $B{\cdot}T$ indicator vectors, so $m \le \min(B{\cdot}T, K)$ and $\mathcal{L}_{\text{div}} \ge K/\min(B{\cdot}T, K)$.*

Proof of Proposition 2. Apply Cauchy–Schwarz on the support: $1 = \left(\sum_{k\in S} \bar{p}_k \cdot 1\right)^2 \le m \sum_{k\in S} \bar{p}_k^2 = m \sum_k \bar{p}_k^2$, so $\sum_k \bar{p}_k^2 \ge 1/m$ with equality iff $\bar{p}_k$ is constant on S; multiplying by K gives the bound. For the second statement, a one-hot $p_{b,t,\cdot}$ places all its mass on a single index, so an average of $B{\cdot}T$ of them is supported on at most $B{\cdot}T$ codes, and on at most K codes in any case. ∎

C Computational Cost

Table A3 times the quantizer's forward and backward pass in isolation at the training configuration, which is the measurement Sec. 7.5 summarizes. At the operating point K=16,384 the quantizer accounts for ~10% of a full optimizer step; LGQ is 3.8× faster and 1.5× smaller than IBQ at matched K, and IBQ exhausts memory at K=262,144 where LGQ still trains. FSQ is cheaper than either and K-independent, since it has no codebook to score against.

End-to-end the differences compress further. Aggregating `sacct` elapsed time over every training job and dividing by epochs completed, one epoch costs 15–17 GPU-hours for *every* quantizer at

Table A3 | Quantizer compute cost. Forward+backward of the quantizer alone at the training configuration (batch 32, T=256, C=64). At our operating point K=16,384 the all-code soft assignment is ~10% of a training step, and LGQ is 3.8× faster and 1.5× smaller than IBQ at every K; IBQ exhausts memory at K=262,144 where LGQ still trains. FSQ is essentially free and K-independent, as it maintains no codebook. Costs grow linearly in K for all codebook-based methods, so we scope the efficiency claim to $K \leq 65{,}536$.

Method	K	img/s	ms/iter	Peak mem (MB)	Share of step
FSQ	16,384	77,093	0.42	1.5	<1%
FSQ	65,536	71,933	0.45	2.3	<1%
SimVQ	16,384	13,123	2.44	856	2.0%
SimVQ	65,536	3,720	8.60	3,196	7.2%
IBQ	16,384	694	46.09	4,700	38.5%
IBQ	65,536	176	182.19	18,548	152%
IBQ	262,144			out of memory	
LGQ	4,096	8,925	3.59	854	**3.0%**
LGQ	16,384	2,627	12.18	3,165	**10.2%**
LGQ	65,536	679	47.10	12,405	39.4%
LGQ	262,144	168	190.33	49,369	159%

K=16,384 (LGQ 16.4, IBQ 17.3, SimVQ 15.4, RotVQ 15.6, LFQ 16.8), and LGQ's own cost rises only from 15.9 to 17.3 GPU-hours per epoch as K grows 16× from 4096 to 65,536, an 8.9% increase for a 16× larger codebook. The two measurements are consistent because the quantizer is a minority of a step at both ends of that range: growing its own share from 3.0% to 39.4% of a step leaves data loading, the perceptual and adversarial losses, and per-epoch validation dominating wall-clock, so a large relative change in the quantizer lands as a single-digit end-to-end one. These aggregate figures are *upper bounds*: jobs are terminated at a wall-time limit mid-epoch, so each carries a partial epoch of unattributed work, and they mix training with per-epoch validation.

D Baseline Quantizers in Detail

This section expands the per-method summary of Sec. 5. Table A4 contrasts the collapse fix each baseline adopts, the structural cost of that fix, and the corresponding LGQ design decision; the paragraphs below give the detail behind each row.

VQ-VAE [1]. The original encoder–quantizer–decoder template uses the argmin assignment with a straight-through estimator [8]: only the selected code $k^\star$ receives a codebook gradient, so non-selected codes drift into *dead codes*, which is the well-known codebook collapse [2, 5]. EMA updates and codebook resets mitigate but do not structurally fix. LGQ avoids this by giving every code a gradient through the soft-assignment.

RotVQ / Rotation Trick [7]. Fifty et al.keep the argmin forward pass but replace the straight-through gradient with a rotation and rescaling that aligns the encoder output with its assigned code, preserving their relative angle and magnitude and recovering information that vanilla STE

Table A4 | Theoretical contrast of the five baselines and LGQ. Each row identifies the baseline's failure-mode fix, the structural cost of that fix, and the corresponding LGQ design decision that avoids the cost. The SimVQ row characterizes the published single-linear reparameterization; our experiments use a stronger nonlinear MLP variant (Sec. F).

Method	Collapse fix	Structural cost	LGQ's response
VQ	EMA + argmin assignment	Only k^* learns; dead codes accumulate	Soft-assignment ⇒ all codes learn
RotVQ	Rotation+rescale STE	Argmin still picks one code per step	Soft gradient avoids needing a rotation
FSQ	Fixed scalar grid	No data adaptivity, axis-aligned	Learnable codes in $\mathbb{R}^C$
SimVQ	Shared linear basis $\mathbf{W}$	Codebook constrained to one learned subspace	Per-code degrees of freedom retained
IBQ	Softmax STE on all codes	Unnormalized dot-product logits, fixed τ, entropy penalty	ℓ_2^2 logits, annealed τ, soft-distribution regularizers

discards. The fix acts on the gradient path rather than on how many codes are learned per step, since the assignment remains hard. LGQ sidesteps this entirely: its soft-assignment gradient already gives every code a signal proportional to its proximity, with no gradient surgery.

FSQ [6]. Finite Scalar Quantization replaces the learned codebook with a Cartesian product of fixed scalar levels per channel (e.g., $L = [8, 8, 8, 8, 4]$ at K=16,384). Collapse is eliminated by construction, every grid point is a valid code, so the codebook cannot drift because it cannot move at all, but the trade-off is a loss of adaptivity: the axis-aligned grid cannot concentrate capacity in high-density regions or adapt to off-axis cluster structure. LGQ keeps a flat learnable codebook while inheriting FSQ's collapse-resistance via regularization, gaining adaptivity and full utilization at once.

LFQ [4]. Lookup-Free Quantization (MAGVIT-v2) encodes each spatial location as a per-channel sign pattern, giving an implicit codebook of 2^d entries kept balanced by an entropy loss. It scales the implicit codebook well, but restricts codes to the corners of the $\{-1, +1\}^d$ hypercube, so it cannot place cluster codes off the sign-pattern lattice. On our backbone this shows up as a slow start: one of the 14 binary dimensions saturates to a constant sign from epoch 8, halving the reachable codebook to ∼50% utilization, and the dimension only unsaturates at epoch 25, after which utilization holds at 100%. This is why LFQ is reported at a later epoch than the other rows of Table 2. LGQ's codes sit anywhere in $\mathbb{R}^C$, removing the lattice constraint while keeping the entropy-style "every code should be used" intuition via $\mathcal{L}_{\text{div}}$.

SimVQ [5]. Zhu et al.reparameterize the codebook through a single shared linear layer $\mathbf{CW}$ ($\mathbf{C}$ a frozen random matrix, $\mathbf{W}$ the only learnable component), so updating $\mathbf{W}$ moves all K codes jointly and VQ's disjoint per-code optimization becomes optimization of one linear subspace. This reaches 100% utilization (reported at K=65,536 on ImageNet), but binds every code to a single learned basis, limiting the geometry the codebook can represent. LGQ instead keeps per-code degrees of

freedom, with each $\mathbf{e}_k$ independently learnable, while still preventing collapse through explicit usage regularization (λ_{peak}=λ_{div}=0.005 throughout the K sweep, held fixed rather than retuned per K).

VQGAN-LC [12]. Closest in spirit to our K-scaling experiments, VQGAN-LC scales the codebook to 100,000 entries at 99% utilization by freezing a codebook initialized from pretrained visual features and learning only a projector onto it. Like SimVQ it buys utilization by giving up per-code adaptivity, and like SimVQ its codes cannot migrate toward the encoder distribution. We do not run it: it requires a pretrained feature extractor to initialize the codebook, which breaks the matched-backbone, matched-initialization protocol under which every row of Table 2 is produced, so a number from it would not be comparable to the others.

IBQ [9]. Index Backpropagation Quantization is closest to LGQ in spirit: it applies STE to a softmax over $\mathbf{z}^\top \mathbf{C}_k$ logits, so every code receives a gradient proportional to its probability, exactly as LGQ does, and scales to K=2^{18} with a two-sided (double) quantization loss, a deeper backbone, and a MAGVIT-v2-style entropy penalty. It differs from LGQ in three ways: unnormalized dot-product logits instead of squared ℓ_2; a fixed softmax temperature instead of annealed $\tau : 1.0 \to 0.1$; and an entropy penalty on the assignment distribution instead of peakedness+diversity on the soft-assignment. Sec. 5.2 isolates the first of these three differences experimentally.

E Training Setup

We train on ImageNet ILSVRC-2012 [16] at 256×256 on H100 GPUs using DDP across 4 GPUs, with per-GPU batch size 8 (global batch 32). The optimizer is AdamW [17] with a learning rate 3×10^{-4}, no schedule, AMP enabled. Data augmentation: random resized crop and horizontal flip with ImageNet normalization.

F Quantizer Configurations

LGQ: $\lambda_{\text{peak}} = \lambda_{\text{div}} = 0.005$, τ annealed linearly $1.0 \to 0.1$, flattened mode (one codebook of size K over C-dimensional vectors, not per-channel), squared-ℓ_2 distance, fixed-seed random initialization of code vectors. **RotVQ**: argmin assignment with the rotation trick [7] enabled, commitment weight β=1.0, and EMA codebook updates with decay γ=0.8. **FSQ**: level configuration $[8, 8, 8, 8, 4]$, giving $K = 16{,}384$. **SimVQ**: the shared-codebook reparameterization of [5] with the codebook transform set to a Linear–ReLU–Linear MLP (hidden width $2C$) and commitment weight $\beta = 10$. This is a deliberate deviation from the published single-linear map (**CW**, β=1.0): the more expressive MLP transform improved SimVQ's reconstruction on our backbone, so we report this stronger variant rather than understate the baseline. **IBQ**: as published [9]: unnormalized dot-product logits $\mathbf{z}^\top \mathbf{e}_k$ into the softmax, a fixed softmax temperature, and the standard entropy-based assignment loss.

G Compute Budget and Convergence

Each tokenizer training run uses 4×H100 80 GB on an internal university GPU cluster with a wall-time budget of two days per resume; long runs are checkpointed and resumed. MaskGIT training

runs use a single H100 (or 4×H100 in DDP for the longer runs). Total compute across all tokenizer and MaskGIT runs reported here is on the order of a few thousand H100 GPU-hours.

Convergence of the comparison. Each tokenizer is reported at its best-validation-rFID checkpoint, by which point the per-epoch rFID curves have plateaued; the differing checkpoint epochs reflect differing convergence rates under a shared wall-clock budget, not unequal training effort, so a later-epoch checkpoint is not a better-resourced one. Each MaskGIT transformer is likewise trained to its validation-NLL plateau, and the two weakest baselines (RotVQ, SimVQ) reach fewer epochs but remain far behind the leaders, so their stopping point does not affect the ranking.

H Selection-Rule Analysis: ℓ_2^2 vs. Dot-Product (vs. IBQ)

This section gives the propositions, proofs, and probabilistic interpretation behind the ℓ_2^2-vs-dot-product comparison of Sec. 5.2. The closest baseline to LGQ at the algorithmic level is IBQ [9], which also applies STE to a softmax over all K codes but scores codes by the unnormalized inner product $\langle \mathbf{z}, \mathbf{e}_k \rangle$ rather than by squared ℓ_2 distance.

Proposition 3 | Bayes-optimality of nearest-neighbour selection. *For a fixed codebook $\{\mathbf{e}_k\}_{k=1}^K$ and an MSE reconstruction loss $\|\hat{\mathbf{z}} - \mathbf{z}\|_2^2$, the deterministic hard quantizer that minimizes the expected reconstruction error is*

$$q^\star(\mathbf{z}) \;=\; \arg\min_k \|\mathbf{z} - \mathbf{e}_k\|_2^2.$$

Proof. For any quantizer q, $\mathbb{E}\|\mathbf{z} - \mathbf{e}_{q(\mathbf{z})}\|^2 = \mathbb{E}\,\min_k \|\mathbf{z} - \mathbf{e}_k\|^2 + \mathbb{E}\big(\|\mathbf{z} - \mathbf{e}_{q(\mathbf{z})}\|^2 - \min_k \|\mathbf{z} - \mathbf{e}_k\|^2\big)$; the second term is non-negative and vanishes exactly when $q = q^\star$ (Lloyd's optimality condition [18]). ∎

LGQ's forward pass is $k^\star = \arg\max_k p_{b,t,k} = q^\star(\mathbf{z}_{b,t})$ because p_k is monotone in $-\|\mathbf{z} - \mathbf{e}_k\|^2$, so LGQ's selection rule is Bayes-optimal *for latent-space MSE*, whereas IBQ's is not. Proposition 3 concerns the latent reconstruction term $\|\hat{\mathbf{z}} - \mathbf{z}\|_2^2$ in isolation, for a *fixed* codebook. LGQ is trained under a composite objective that additionally includes pixel, perceptual, adversarial, peakedness, and diversity terms, over a codebook that is itself being learned. The proposition therefore motivates the Euclidean assignment rule; it does not establish that LGQ is optimal for the objective actually optimized, and it does not by itself account for the observed rFID gap with IBQ. We treat that gap as empirical evidence, isolated by the matched logit ablation rather than by this proposition.

Corollary 4 | Norm-bias gap of dot-product selection. *Let $q_{\mathrm{IBQ}}(\mathbf{z}) = \arg\max_k \langle \mathbf{z}, \mathbf{e}_k \rangle$. Because $\arg\min_k \|\mathbf{z} - \mathbf{e}_k\|^2 = \arg\min_k\big(-2\langle \mathbf{z}, \mathbf{e}_k \rangle + \|\mathbf{e}_k\|^2\big)$, $q_{\mathrm{IBQ}} = q^\star$ iff $\|\mathbf{e}_k\|^2$ is constant in k. When codebook norms vary, q_{IBQ} systematically prefers larger-norm codes, with per-sample excess distortion*

$$\begin{aligned}\Delta_{\mathrm{IBQ}}(\mathbf{z}) \;&=\; \|\mathbf{z} - \mathbf{e}_{q_{\mathrm{IBQ}}(\mathbf{z})}\|^2 - \|\mathbf{z} - \mathbf{e}_{q^\star(\mathbf{z})}\|^2 \\ &\le\; \max_{j,k}\big(\|\mathbf{e}_j\|^2 - \|\mathbf{e}_k\|^2\big).\end{aligned}$$

Proof. Expanding $\|\mathbf{z} - \mathbf{e}_k\|^2 = \|\mathbf{z}\|^2 - 2\langle \mathbf{z}, \mathbf{e}_k \rangle + \|\mathbf{e}_k\|^2$, q_{IBQ} matches $q^\star$ when the $\|\mathbf{e}_k\|^2$ term is constant in k and is otherwise dominated by the largest-norm code. The substitution error is bounded by the maximum norm gap. ∎

Table A5 | Codebook norms in the dot-product arm. The same 3,300 codes are dead at every checkpoint. Dead and live norms are disjoint, and the separation widens monotonically: every one of the 3,300 shrank between consecutive checkpoints, and the dead ranges at successive epochs do not overlap. The last column is the correlation between a code's norm and the log of its usage.

Epoch	Dead	Dead $\|\mathbf{e}_k\|$	Live $\|\mathbf{e}_k\|$	corr
18	3,300	[0.823, 1.022]	[5.273, 5.909]	+0.996
20	3,300	[0.648, 0.805]	[5.232, 5.859]	+0.997
22	3,300	[0.511, 0.634]	[5.198, 5.848]	+0.997

Nothing in IBQ's loss explicitly constrains $\|\mathbf{e}_k\|$ to be constant across k, so the gap is a structural property of the rule rather than an implementation detail, and it grows with codebook heterogeneity. The bound above is worst-case over $\mathbf{z}$ and independent of it, so it upper-bounds rather than predicts the typical excess distortion. The realized effect is therefore measured rather than derived, by the matched logit ablation of Sec. 5.2, which holds backbone, seed, schedule and budget fixed and varies only the quantity the softmax scores. That experiment, rather than the IBQ–LGQ rFID gap of Table 2, is what isolates the selection rule: the published IBQ configuration also differs in its two-sided (double) quantization loss, backbone depth, and entropy penalty.

Measured norm bias. The matched logit ablation lets us test Corollary 4 directly, since both arms share a seed and an initialization and differ only in the scored quantity. We identify dead codes by the batch-averaged soft assignment $\bar{p}_k$ stored by the quantizer; in the dot-product arm this statistic is bimodal, around 10^{-14} against above 10^{-6}, so a 10^{-8} cut is insensitive to its exact value. The dead set is fixed from the fourth epoch onward and never changes membership (Table A5).

This is the mechanism of Corollary 4 operating during training rather than at a fixed codebook: a code that falls behind on magnitude loses assignments, and without assignments it receives no gradient with which to grow back, so the norm gap compounds. The ℓ_2^2 arm is the control. It has no dead codes at all, and the correlation between norm and usage is +0.08, despite carrying a 3.6× spread in norm ($\|\mathbf{e}_k\| \in [2.80, 10.01]$ at epoch 23). The squared distance therefore does not require a norm-homogeneous codebook to use every code, whereas the inner product does. We report this as a property of these two runs; it is a measurement of the predicted effect, not an independent proof of the corollary.

Scaling across K: does the baseline degrade where LGQ does not?. A natural question about the codebook-size scaling of Sec. 7.2 is whether it is specific to LGQ or would appear for any quantizer. We therefore trained IBQ at K=4,096, 16,384 and 65,536 under the identical protocol (same encoder, decoder, discriminator, optimizer, effective batch, seed and data order) and compare at a *matched* epoch at each K, taking every value from the training-time validation log of the run itself. Table A6 reports epoch 24 at K=4,096, epoch 45 at K=16,384 and epoch 23 at K=65,536, in each case the deepest epoch at which both runs have a logged rFID.

LGQ is ahead on every metric at all three codebook sizes, so the scaling behaviour is not an artifact of how we selected checkpoints for Table 3. The utilization column is the more informative one, because the two methods move in opposite directions as training proceeds. At K=4,096 and

Table A6 | LGQ against IBQ across codebook size, at matched budget. The two methods are compared at a matched epoch at each K: epoch 24 at K=4,096, epoch 45 at K=16,384, and epoch 23 at K=65,536. Every value is read from the run's own training-time validation log, so the two methods are measured identically. Neither method has reached the budget of the main comparison at these epochs and both continue to improve, so these are budget-matched rather than final numbers, and they are not comparable to the main-paper tables. Each row is a single training run. "Active" is the number of codes assigned at least once. **Bold** marks the better of the two at each K.

K	Method	PSNR ↑	SSIM ↑	LPIPS ↓	Rec. loss ↓	rFID ↓	Util. %	Active
4,096	IBQ	21.190	0.5999	0.2852	0.2437	20.68	99.95	4,094
	LGQ	**21.409**	**0.6065**	**0.2705**	**0.2343**	**16.14**	**100.00**	**4,096**
16,384	IBQ	21.492	0.6162	0.2614	0.2351	16.63	99.56	16,312
	LGQ	**21.738**	**0.6260**	**0.2436**	**0.2230**	**12.89**	**100.00**	**16,384**
65,536	IBQ	21.637	0.6158	0.2593	0.2279	16.10	55.07	36,088
	LGQ	**22.093**	**0.6460**	**0.2304**	**0.2145**	**10.82**	**96.84**	**63,465**

K=16,384 IBQ holds 99.95% and 99.56%, so its anti-collapse machinery works at small and medium K. At K=65,536 it peaks at 98.18% in epoch 2 and then falls, to 67.33% by epoch 12 and 55.07% (36,088 of 65,536 codes) by epoch 23. LGQ at the same K moves the other way, rising from 82.64% at epoch 12 through 89.96% at epoch 18, 95.19% at epoch 22 and 96.84% at epoch 23 to 100% at epoch 29, after which it stays there. IBQ therefore begins with the higher utilization and ends far below it: the two curves cross. We read this as the entropy penalty being tuned at a codebook size where its temperature is appropriate, and losing codes once K grows past that point, whereas LGQ's diversity term is stated relative to K and needs no retuning. We report this from one run per cell and do not claim it generalizes to other baselines, which we have not swept across K.

Probabilistic interpretation. LGQ's softmax over $-\|\mathbf{z} - \mathbf{e}_k\|^2/\tau$ is the posterior of a Gaussian mixture with isotropic covariance $\sigma^2 = \tau/2$; the hard-assignment rule is therefore MAP inference under a GMM. IBQ's softmax over $\langle \mathbf{z}, \mathbf{e}_k \rangle$ is the posterior of a von Mises–Fisher mixture on the hypersphere, which is the correct model *only* when features are constrained to unit norm. Since both our backbone and IBQ's produce unconstrained $\mathbb{R}^C$ features and the decoder consumes Euclidean inputs, the vMF prior is misspecified for both the data and the loss, so LGQ's distance-based scoring matches the geometry that the rest of the pipeline actually uses. The dot-product rule does have two genuine advantages, so the trade-off is worth stating explicitly: its encoder gradient $\partial\langle \mathbf{z}, \mathbf{e}_k \rangle/\partial \mathbf{z} = \mathbf{e}_k$ is bounded by $\|\mathbf{e}_k\|$ (whereas $\partial\|\mathbf{z} - \mathbf{e}_k\|^2/\partial \mathbf{z} = 2(\mathbf{z} - \mathbf{e}_k)$ grows with the encoder–codebook gap), and it costs a single BLAS matmul without the squared-difference expansion. LGQ accepts both costs, spending an annealed τ to keep the backward signal bounded (the annealed-softmax Jacobian bound of Sec. 4.1) and paying for a pairwise distance, in exchange for a selection rule whose forward pass is the Bayes-optimal nearest-neighbour rule for the reconstruction loss (Proposition 3). The controlled isolation of the selection rule alone is the matched logit ablation of Sec. 5.2.

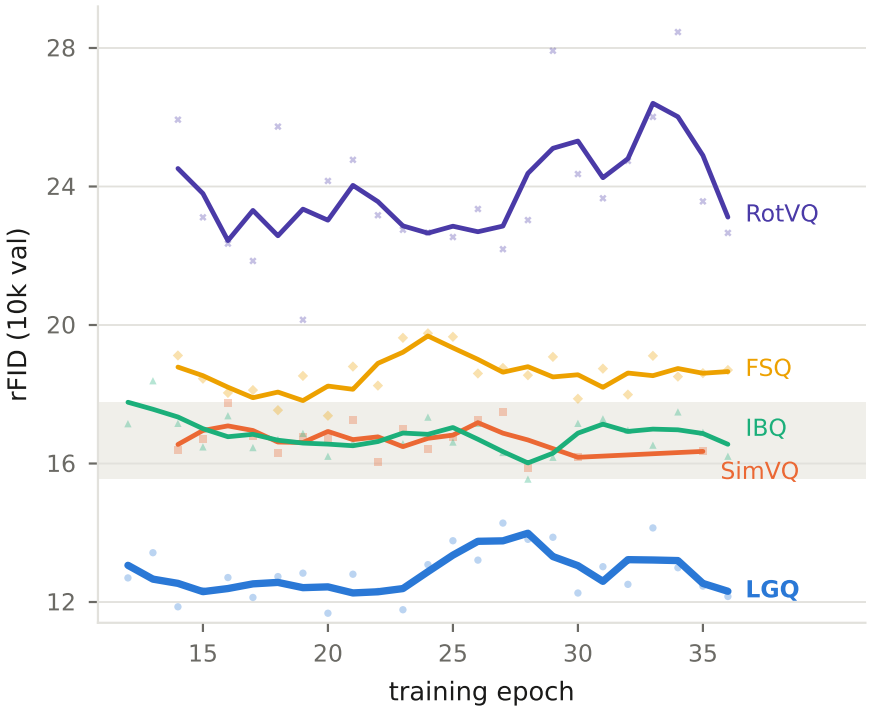


Figure A1 | Reconstruction rFID against training epoch for five of the six quantizers of the main comparison over the epochs. Lines are a three-epoch centred rolling mean and the faint markers are the raw per-epoch values. Of the 10 method pairs measured at four or more common epochs, 9 never invert. LGQ leads at all 13 epochs at which it was measured in this window, with a minimum margin of 1.74 rFID (mean 3.36) against a within-run epoch-to-epoch scatter of 0.60. rFID here is the 10,000-image validation metric logged during training; the 50,000-image ordering was checked separately at two checkpoints and was identical at both.

I Per-Epoch Reconstruction Curves

Every tokenizer in this paper is reported at its best-validation-rFID checkpoint under a shared wall-clock budget, so the natural question is whether the ordering those checkpoints induce is an artifact of where each run happened to be stopped. Fig. A1 plots validation rFID against training epoch for five of the six quantizers of the main comparison, one run each, over a common window ending at epoch 36, and answers it directly: the curves separate into three bands that never meet, and the single pair that does exchange places is a pair whose runs are not separable at this budget.

J Codebook Geometry

Fig. A2 gives the UMAP overlay of the learned codes on the continuous encoder distribution referenced in Sec. 7.3; the tokenizer architecture is diagrammed in Fig. 2.

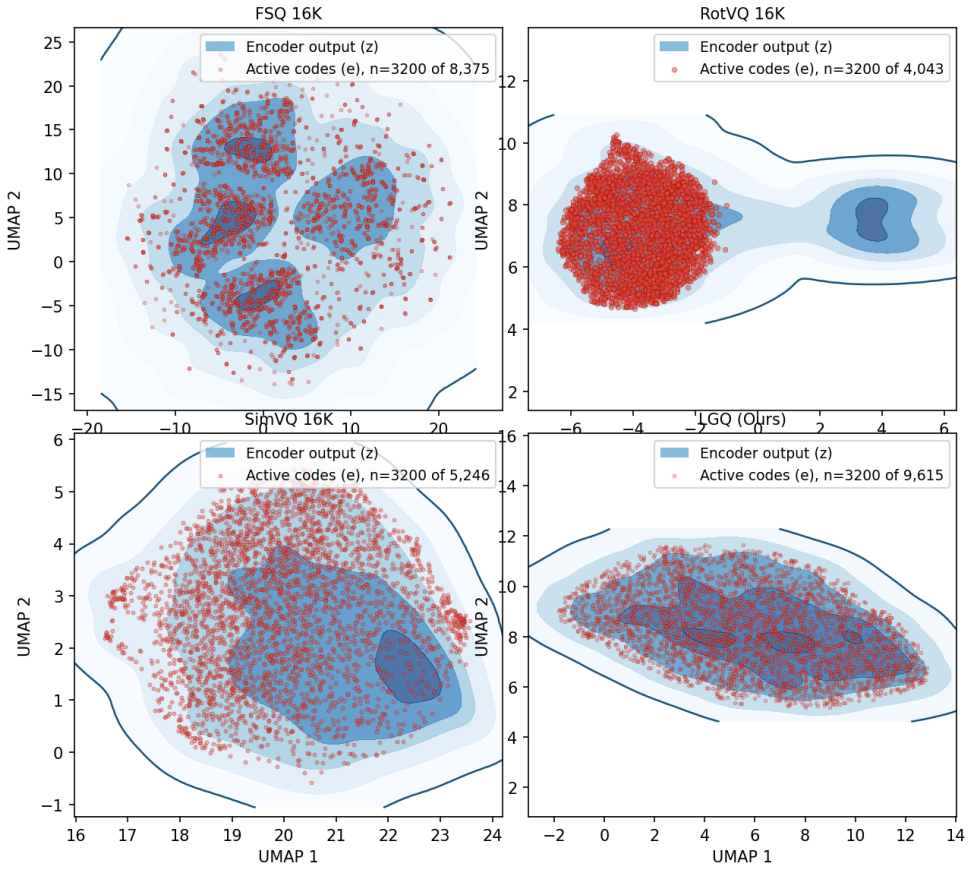


Figure A2 | UMAP of encoder latents (gray) with active codebook entries (red) at $K = 16{,}384$. LGQ's codes spread across the encoder distribution, consistent with its 100% utilization; RotVQ's cluster in a small subregion, the visual signature of under-utilization. The overlay geometry tracks the rFID ranking of Table 2.

K Full Qualitative Comparisons

Figs. A3 and A4 give the complete qualitative grids summarized by the single-example strip in Fig. 3. Fig. A3 shows reconstructions of six ImageNet validation images for every tokenizer; Fig. A4 shows class-conditional MaskGIT samples across eight ImageNet classes for all five tokenizers. Because the cells of Fig. A4 are chosen by classifier confidence, we also give Fig. A5, an unselected draw in which every generated sample is shown; the two together separate what the tokenizers can produce from what they typically produce.

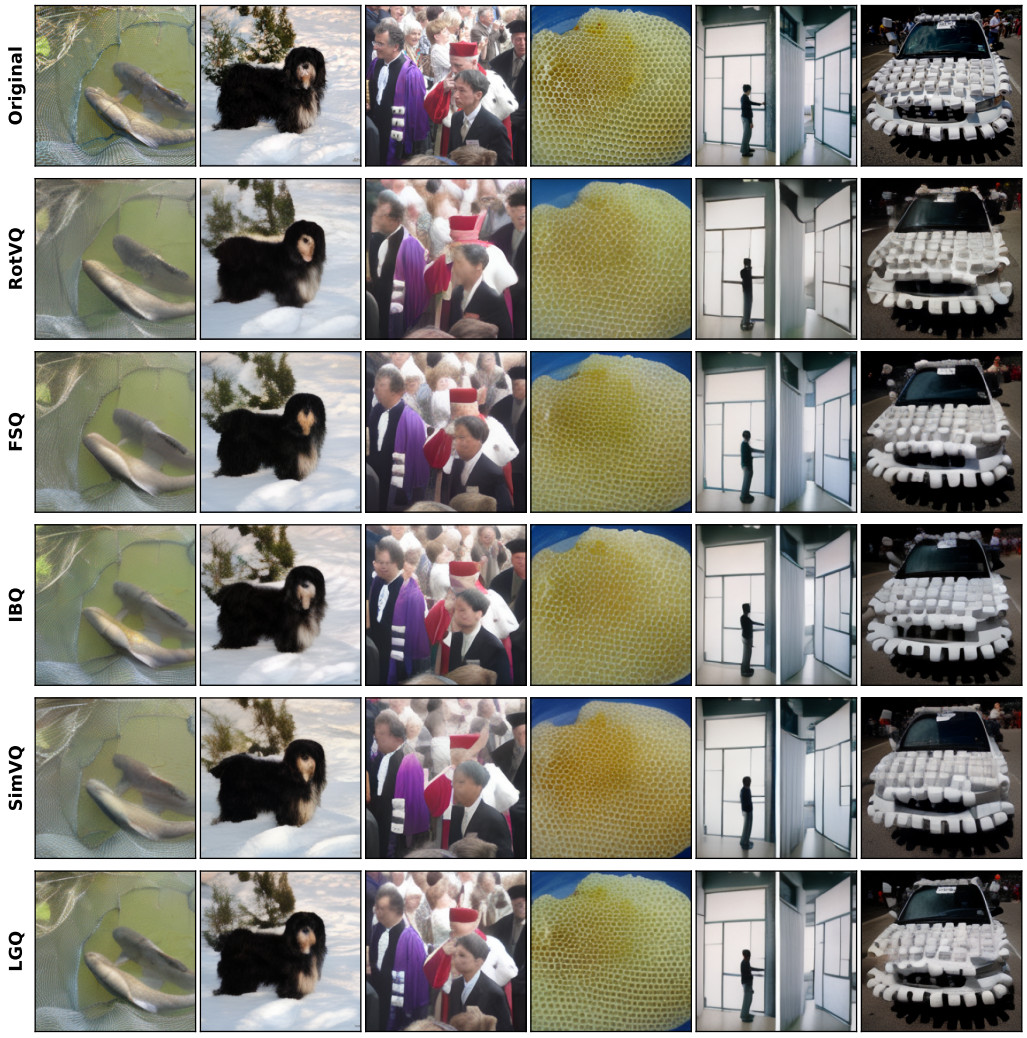


Figure A3 | Reconstructions at K=16,384 (full grid). Top row: original ImageNet validation images; each subsequent row is the reconstruction from one frozen tokenizer (best-rFID checkpoint). LGQ's reconstructions retain the finest texture and color fidelity, consistent with its best rFID/LPIPS in Table 2; RotVQ shows the most blur, matching its higher rFID. Fig. 3 uses the second column (dog) of this grid.

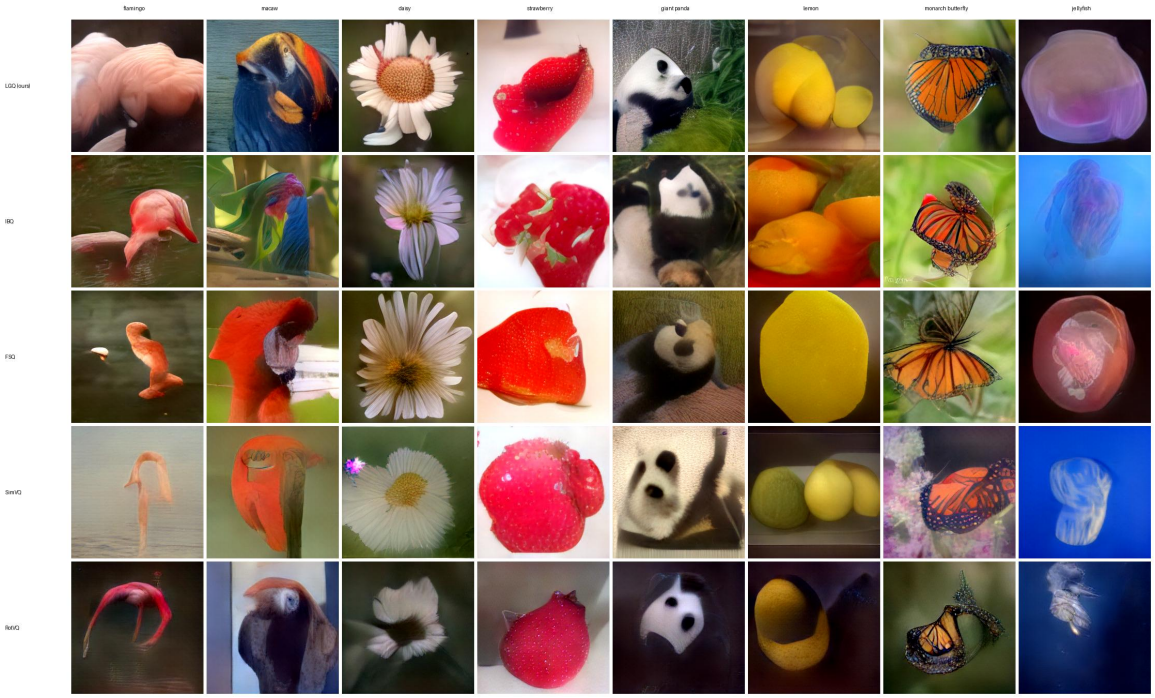


Figure A4 | Class-conditional MaskGIT samples at K=16,384 (full grid). Rows are tokenizers ordered by gFID (top to bottom: LGQ, IBQ, FSQ, SimVQ, RotVQ), columns are eight ImageNet classes; each cell is the highest-confidence generation for that class, selected by a pretrained ImageNet classifier from the same 50K-sample set used for the gFID in Table 5. Quality degrades top to bottom in step with gFID: LGQ's samples are the sharpest and most class-coherent (best gFID 57.69), while RotVQ's are the blurriest (gFID 92.99). Because the cells are classifier-selected, this grid is a best-case display for every method; Fig. A5 shows an unselected draw.

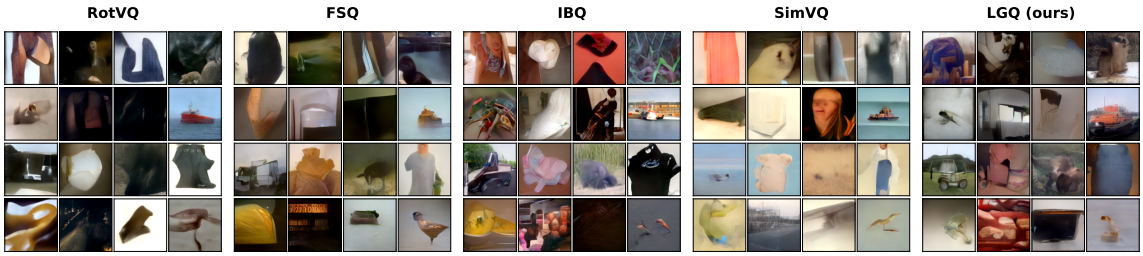


Figure A5 | Unselected class-conditional MaskGIT samples at K=16,384. Panels are the same five tokenizers as Fig. A4 (left to right: RotVQ, FSQ, IBQ, SimVQ, LGQ), sixteen samples each. Class labels are drawn uniformly at random under a fixed seed and reused across panels, so each grid position holds the same class in every panel. *Every generated sample is shown*: no filtering and no confidence ranking, in contrast to Fig. A4. At this backbone scale (27.99M encoder/decoder parameters, 2.56× smaller than the standard VQ-GAN $f16$) absolute sample quality is low for every method and only RotVQ is clearly distinguishable by eye, consistent with its gFID of 92.99; LGQ, IBQ, SimVQ and FSQ are not visually separable at this draw. We therefore rest the generation comparison on the gFID over 50K samples of Table 5 rather than on qualitative panels.