LEARNED IMAGE COMPRESSION WITH VECTOR QUANTIZATION-BASED PRIOR MODELING

Anonymous authors
Paper under double-blind review

ABSTRACT

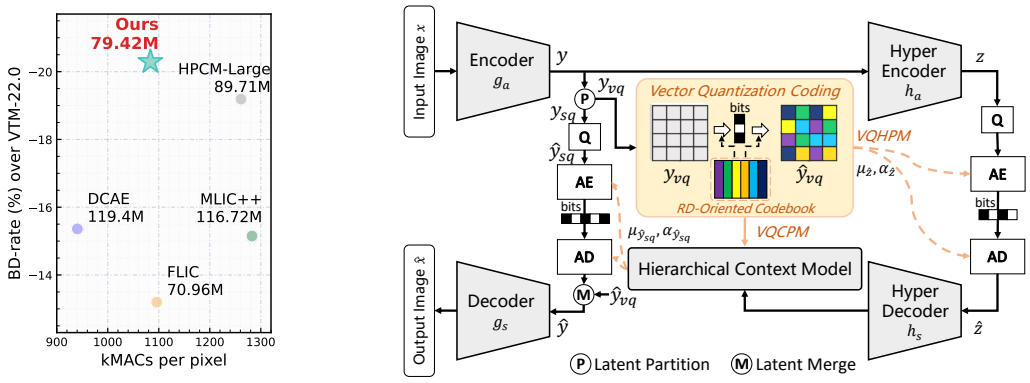
Entropy modeling is essential for learned image compression (LIC), where the latents are usually scalar-quantized and subsequently encoded based on estimated priors. Compared to scalar quantization (SQ), vector quantization (VQ) is theoretically more efficient for compression performance. However, the codebook size in VQ grows exponentially with the bitrate, resulting in an extremely high-dimensional optimization landscape. Such a complex landscape poses significant challenges to convergence and often leads to severe codebook collapse, making it difficult to achieve effective rate-distortion (RD) optimization. To make VQ tractable within an end-to-end framework, we propose Vector Quantization-Based Prior Modeling (VQPM). VQPM applies VQ to a subset of the latents with the lowest spatial resolution. These low-resolution features inherently capture the image's global structure and are coded first, without any decoded context. VQ is particularly suitable here as it can model the dependencies within the vector itself, compensating for the lack of external context. The vector-quantized latents then serve as global structural guidance for the subsequent hyperprior and contextual prior modeling. We further introduce an RD-oriented codebook training strategy that optimizes the codebook using end-to-end RD gradients. Experimental results show that VQPM achieves state-of-the-art RD performance, reducing PSNR BD-rate over VTM-22.0 by 20.29%.

1 INTRODUCTION

Lossy image compression is a rate-distortion optimization problem that seeks to minimize the bitrate under a given reconstruction distortion. Classical codecs, including JPEG (Wallace, 1991), JPEG2000 (Skodras et al., 2001), BPG (Bellard, 2015), and VVC (Bross et al., 2021), rely on hand-crafted transform, prediction, quantization, and entropy coding. Recently, learned image compression (LIC) addresses this problem with an end-to-end optimized transform coding pipeline and has achieved competitive RD performance (He et al., 2022; Liu et al., 2023; Jiang & Wang, 2023; Li et al., 2024; Lu et al., 2025; Li et al., 2025; Chen et al., 2026a). In most LIC models, an image is mapped to compact latent representations, which are then scalar-quantized (SQ) and entropy-coded using priors estimated by an entropy model.

Theoretically, vector quantization (VQ) (Gersho & Gray, 1992; Lookabaugh & Gray, 1989) is generally more efficient than scalar quantization (SQ) in terms of compression performance. By jointly quantizing multiple latent dimensions, VQ can better exploit inter-dimensional dependencies and benefit from the space-filling properties of vector quantization cells (Lookabaugh & Gray, 1989; Conway & Sloane, 1982). Meanwhile, the recent success in VQ-based autoregressive generative models, including VQ-GAN (Esser et al., 2021), MaskGIT (Chang et al., 2022), MAGVIT (Yu et al., 2023), and VAR (Tian et al., 2024), demonstrates that discrete VQ tokens possess strong representational capacity. These properties suggest that vector quantization (VQ) holds promise for learned image compression (LIC). However, directly using VQ as the latent quantizer in end-to-end training remains difficult because the codebook size grows exponentially with the vector dimension and bitrate. For an n-dimensional vector quantized at R bits per dimension, the number of codewords scales as 2^{nR}, resulting in an extremely high-dimensional optimization landscape that poses significant challenges to convergence and often leads to severe codebook collapse. To make VQ more tractable, existing VQ-based learned image compression (LIC) methods restrict the optimization landscape through various strategies, such as applying VQ only at ultra-low bitrates (Mao et al., 2024;

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.



(a) (b)

Figure 1: **(a)** PSNR BD-rate over VTM-22.0 vs. end-to-end complexity on Kodak, compared with recent LIC methods (Jiang & Wang, 2023; Li et al., 2024; Lu et al., 2025; Li et al., 2025); upper-left is better. **(b)** Overall VQPM framework, including VQ-based HyperPrior Modeling (VQHPM) and VQ-based Contextual Prior Modeling (VQCPM). AE/AD: arithmetic encoder/decoder; Q: quantization; P/M: latent partition/merge.

Kong et al., 2024) or constraining quantization cells with lattice-structured codebooks (Zhang & Wu, 2023; 2024).

To exploit the potential of vector quantization (VQ) within existing advanced learned image compression models while avoiding its optimization difficulties, we propose Vector Quantization-Based Prior Modeling (VQPM). Instead of applying VQ to the entire latent representation, VQPM uses it only on a small subset of low-resolution latents that exhibit weak spatial correlation. These features capture the global structure of the image and are encoded first, without access to any previously decoded context. VQ is especially well-suited for this setting, as it can model internal dependencies within each vector to compensate for the absence of external contextual information. The remaining latents are handled by standard scalar quantization and autoregressive context modeling. To fully exploit the VQ latents, we use them to guide subsequent entropy modeling in two ways. We propose Vector Quantization-Based Hyperprior Modeling (VQHPM) to enable content-adaptive hyperpriors, and Vector Quantization-Based Contextual Prior Modeling (VQCPM) to incorporate the VQ latents into the hierarchical progressive coding framework (Li et al., 2025) for improved contextual modeling accuracy.

The training strategy for vector quantization is crucial in our framework. Unlike VQ-based generative models, where codebook learning relies mainly on reconstruction, our method directly links the selected codeword to both bitrate and distortion through entropy modeling. To better leverage VQ's potential, we propose an RD-oriented codebook training strategy. Each codeword is decomposed into a base component updated by exponential moving average (EMA) to ensure stable assignments and an RD-oriented bias optimized end-to-end with the RD loss to improve the rate-distortion trade-off.

As shown in Fig. 1a, our method achieves a state-of-the-art trade-off between compression performance and computational complexity. Our contributions can be summarized as follows:

- We propose Vector Quantization-Based Prior Modeling (VQPM) for learned image compression. VQPM vector-quantizes a subset of first-coded low-resolution latent features and uses the resulting VQ latents as structured guidance for entropy modeling.
- We propose an RD-oriented VQ training strategy. Each codeword is decomposed into an EMA-updated base codeword and an RD-oriented bias, enabling RD-aware training under the end-to-end RD loss.
- Our method achieves state-of-the-art rate-distortion performance, reducing PSNR BD-Rate over VTM-22.0 by 20.29%, 19.94%, and 23.95% on Kodak, CLIC Pro Valid, and Tecnick, while preserving a favorable trade-off between performance and computational complexity.

2 Related Work

Learned Image Compression. Learned image compression (LIC) has achieved remarkable progress through end-to-end rate-distortion optimization, with many studies improving transform architectures using deeper CNNs, attention mechanisms, and other advanced modules (Chen et al., 2021; Cheng et al., 2020; He et al., 2022; Liu et al., 2023; Jiang & Wang, 2023; Li et al., 2024; Lu et al., 2025;

Li et al., 2025). GLIC (Chen et al., 2026a) further explores graph neural networks for content-adaptive image compression. Entropy modeling is essential in LIC, where most methods estimate latent probabilities using hyperpriors and context models based on parametric distributions such as Gaussian or generalized Gaussian models (Ballé et al., 2018; Minnen et al., 2018; Zhang et al., 2024). Recent studies further improve context modeling from raster-scan autoregression to spatial, channel-wise, and hybrid dependencies (Minnen et al., 2018; He et al., 2021; Minnen & Singh, 2020; He et al., 2022; Qian et al., 2022; Jiang & Wang, 2023; Kim et al., 2024; Chen et al., 2025; Lu et al., 2025; Kuang et al., 2025; Chen et al., 2026b). However, at the earliest coding stage, conditional context is limited and context-based entropy modeling provides only weak priors, motivating additional guidance beyond conventional context modeling.

Vector Quantization in Compression. Vector quantization (VQ) has long been regarded as a more rate-distortion-efficient alternative to scalar quantization, due to its joint quantization capability and space-filling advantage (Lookabaugh & Gray, 1989; Conway & Sloane, 1982). Despite these advantages, practical VQ is limited by storage cost and optimization difficulty, motivating variants such as product VQ, multi-stage VQ, tree-structured VQ, and lattice VQ (Sabin & Gray, 1984; Jegou et al., 2010; Juang & Gray, 1982; Buzo et al., 1980; Gersho, 1979). Recent learned compression methods introduce VQ through lattice-based quantization or entropy-constrained latent-space VQ (Zhang & Wu, 2023; 2024; Xu et al., 2025; Feng et al., 2023). However, these methods still use VQ directly for latent quantization, so the VQ optimization remains coupled with the latent coding process. VQ-VAE-style models have also been explored for ultra-low-bitrate perceptual compression (Mao et al., 2024; Xue et al., 2024; Kong et al., 2024; Jia et al., 2024).

Different from these methods, we revisit VQ from the entropy modeling perspective and apply it only to a subset of low-resolution latents to construct a global structural prior, keeping the VQ optimization landscape tractable.

3 Proposed Methods

3.1 Overview

As illustrated in Fig. 1b, our framework follows the standard transform coding pipeline and incorporates a VQ-based prior modeling module. Given an input image $x \in \mathbb{R}^{3\times H\times W}$, the analysis transform g_a produces latent $y = g_a(x \mid \phi) \in \mathbb{R}^{C\times \frac{H}{16}\times \frac{W}{16}}$, which is split into y_{vq} and y_{sq} as described in Sec. 3.2. The subset y_{vq} is vector-quantized using a codebook, and the resulting VQ indices are encoded into the bitstream, while y_{sq} is scalar-quantized as $\hat{y}_{sq} = Q(y_{sq})$ and entropy-coded by the conditional entropy model. At the decoder, $\hat{y}_{vq}$ and $\hat{y}_{sq}$ are recovered, merged, and fed into the synthesis transform to reconstruct $\hat{x} = g_s(\hat{y}_{vq}, \hat{y}_{sq} \mid \theta)$. Most components follow HPCM (Li et al., 2025), with the blocks after the third downsampling stage replaced by Partial Attention Convolution (PATConv) modules (Huang et al., 2025). Architectural details are provided in Appendix A.

We then introduce Vector Quantization-Based Prior Modeling (VQPM) to estimate the entropy parameters of both the hyperprior latents $\hat{z}$ and the scalar-quantized latents $\hat{y}_{sq}$. A hyper-analysis transform extracts side information $z = h_a(y \mid \phi_h)$, which is quantized as $\hat{z} = Q(z)$. Instead of using a conventional factorized prior, Vector Quantization-Based HyperPrior Modeling (VQHPM) predicts the mean and scale parameters of $\hat{z}$ from $\hat{y}_{vq}$, and the decoded hyperprior features are processed by $h_s(\hat{z} \mid \theta_h)$ for subsequent entropy modeling. For $\hat{y}_{sq}$, Vector Quantization-Based Contextual Prior Modeling (VQCPM) incorporates $\hat{y}_{vq}$ into the hierarchical entropy model (Li et al., 2025) to enhance contextual modeling. Following Zhang et al. (2024; 2025); Li et al. (2025), both $\hat{z}$ and $\hat{y}_{sq}$ are modeled by a generalized Gaussian distribution $\mathcal{N}_{\beta_g}(\mu, \alpha)$, where $\beta_g = 1.5$.

Overall, the proposed VQPM enhances entropy modeling by integrating effective vector-quantized priors into both the hyperprior and sequential context modeling branches. The detailed architecture of VQPM is described in Sec. 3.2. To further align codebook learning with the end-to-end rate-distortion optimization objective and fully exploit the advantages of VQPM, we propose an RD-oriented codebook training strategy, which is presented in Sec. 3.3.

3.2 Vector Quantization-Based Prior Modeling

We revisit VQ in LIC as an entropy-prior modeling tool rather than a full latent quantizer. Since early-coded latents lack sufficient decoded context, conventional context modeling provides only weak priors for them. Thus, this stage needs a prior that can capture global structural dependencies without relying on decoded context. VQ naturally fits this role since it jointly quantizes latent vectors

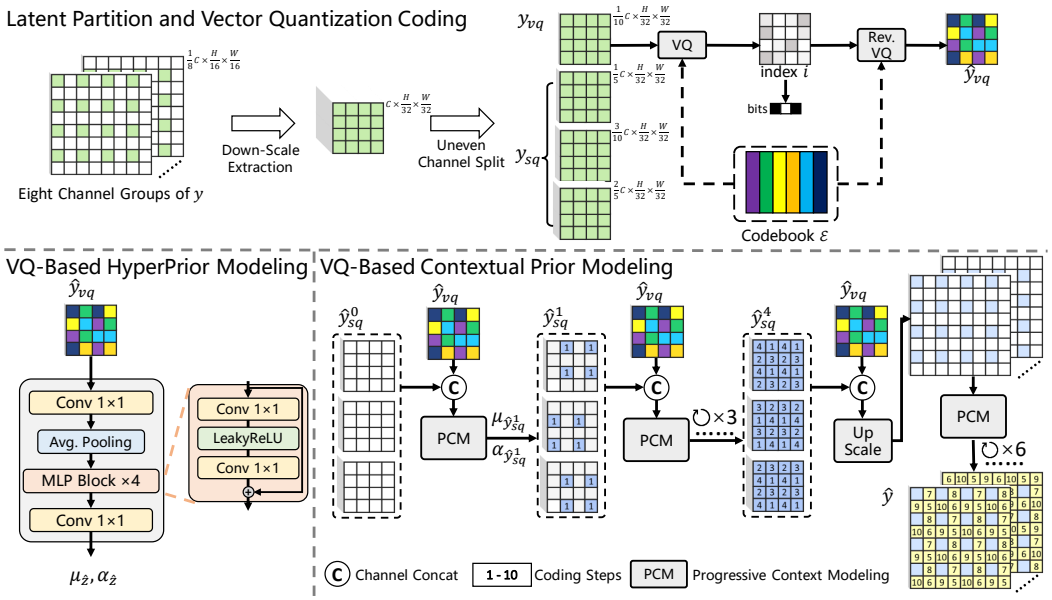


Figure 2: Illustration of the proposed Vector Quantization-Based Prior Modeling (VQPM). **Top** (Latent Partition and Vector Quantization Coding): A low-resolution subset of latents is vector quantized to produce $\hat{y}_{vq}$, while the remaining latents form y_{sq}. Rev. VQ denotes reverse vector quantization. **Bottom left** (Vector Quantization-Based HyperPrior Modeling): VQHPM uses $\hat{y}_{vq}$ to predict the entropy parameters of $\hat{z}$. **Bottom right** (Vector Quantization-Based Contextual Prior Modeling): VQCPM utilizes $\hat{y}_{vq}$ to perform progressive context modeling for entropy estimation.

to exploit intra-vector dependencies without relying on extra context. The resulting quantized latents then provide global structural guidance for subsequent hyperprior and contextual prior modeling through VQHPM and VQCPM.

Latent Partition and Vector Quantization Coding. We begin with latent partitioning and vector quantization coding in VQPM, as illustrated in the top portion of Fig. 2. To avoid the high optimization complexity of applying VQ to the full latent representation, we restrict it to a subset of low-resolution latent features. This design specifically targets the early coding stage, where the elements to be encoded are spatially distant and lack sufficient contextual information. In such scenarios, context modeling is inefficient, and VQ can provide significant benefits.

Following the hierarchical partition design in Li et al. (2025), we divide the latent representation y into eight channel groups and extract the corresponding first-coded low-resolution latents to construct a $2\times$ down-scaled version of y. Within these down-scaled latents, the elements are widely dispersed across the spatial dimension, allowing us to capture more global structure. This representation is therefore effective as guidance for prior modeling. To further control the VQ dimensionality and keep the codebook optimization tractable, we further apply an uneven channel-wise split to these low-resolution latents, resulting in four groups with channel counts proportional to $\frac{1}{10}C$, $\frac{1}{5}C$, $\frac{3}{10}C$, and $\frac{2}{5}C$, respectively. The group with $\frac{1}{10}C$ channels is selected as y_{vq}, while the remaining groups are denoted as y_{sq}.

We apply vector quantization to y_{vq} using a shared codebook at the encoder and decoder. For each input vector, we find the nearest codeword in Euclidean distance and replace the vector with the corresponding codeword embedding. Let $\mathcal{E} = \{\mathbf{e}_k\}_{k=1}^{K}$ denote a codebook containing K codewords, where each $\mathbf{e}_k \in \mathbb{R}^{\frac{1}{10}C}$, and let $\{\mathbf{y}_{vq,t}\}_{t=1}^{M}$ represent the M vector elements extracted from y_{vq}. We then conduct vector quantization on y_{vq} by nearest-neighbor assignment:

$$i_t = \underset{k\in\{1,\dots,K\}}{\arg\min} \|\mathbf{y}_{vq,t} - \mathbf{e}_k\|_2^2, \qquad \hat{\mathbf{y}}_{vq,t} = \mathbf{e}_{i_t}, \tag{1}$$

where i_t is the VQ index of the t-th vector in y_{vq} and $\hat{\mathbf{y}}_{vq,t}$ is the corresponding quantized vector. The full quantized representation $\hat{y}_{vq}$ is obtained by reshaping $\{\hat{\mathbf{y}}_{vq,t}\}_{t=1}^{M}$ back to the original shape.

We encode the VQ indices into the bitstream with a trained categorical entropy model, since this subset is small and no context is available at this stage. Specifically, the probability of each index is

modeled as a categorical distribution:

$$p(i_t = k) = \pi_k, \qquad \pi_k = \frac{\exp(-w_k)}{\sum_{j=1}^{K} \exp(-w_j)}, \tag{2}$$

where $\mathbf{w} = (w_1, \ldots, w_K)$ denotes a set of learnable parameters. The corresponding coding rate is:

$$R_{vq} = \sum_{t=1}^{M} -\log_2 p(i_t). \tag{3}$$

Vector Quantization-Based HyperPrior Modeling. Previous studies usually model $\hat{z}$ with a fully factorized prior, which is content-independent and limited in representation capacity. In this work, we condition the entropy model of $\hat{z}$ on the VQ prior and propose Vector Quantization-Based HyperPrior Modeling (VQHPM).

As illustrated in the bottom left part of Fig. 2, we feed the VQ-quantized latent $\hat{y}_{vq}$ into a lightweight network to predict the entropy parameters of $\hat{z}$:

$$[\mu_{\hat{z}}, \alpha_{\hat{z}}] = \mathrm{Conv}_{1\times1}\Big(\mathcal{B}^4\big(\,\mathrm{GAP}(\mathrm{Conv}_{1\times1}(\hat{y}_{vq}))\big)\Big), \tag{4}$$

where $\mathrm{Conv}_{1\times1}$ denotes a point-wise convolution, $\mathrm{GAP}(\cdot)$ denotes global average pooling over the spatial dimensions, and $\mathcal{B}^4(\cdot)$ denotes four stacked lightweight MLP blocks. Here, GAP extracts a location-free global representation from $\hat{y}_{vq}$ for modeling $\hat{z}$. This design replaces the fixed factorized prior with a VQ-conditioned hyperprior. Consequently, the discrete VQ representation provides more informative global guidance for modeling $\hat{z}$, enabling more accurate entropy estimation.

Vector Quantization-Based Contextual Prior Modeling. After encoding the VQ subset and the VQ-conditioned hyperprior, we introduce Vector Quantization-Based Contextual Prior Modeling (VQCPM) to support hierarchical progressive coding (Li et al., 2025). Different from previous schemes that initialize the early coding stage from the hyperprior, VQCPM further incorporates the VQ prior as additional global structured information, which helps mitigate the lack of contextual prior at the beginning of the coding process.

Let $\hat{y}_{sq}^t$ denote the partially decoded scalar-quantized latent available at coding step t. As shown in the bottom right part of Fig. 2, initial context $\hat{y}_{sq}^0$ is set to zero, and for $t > 0$, $\hat{y}_{sq}^t$ is progressively constructed by filling in the already decoded elements of $\hat{y}_{sq}$ following the coding order specified in Li et al. (2025). At each step, the VQ latent $\hat{y}_{vq}$ is concatenated with $\hat{y}_{sq}^t$ along the channel dimension, and the combined feature is passed through the Progressive Context Modeling (PCM) module (Li et al., 2025) to predict the entropy parameters for the next coding step:

$$[\mu_{\hat{y}_{sq}^{t+1}}, \alpha_{\hat{y}_{sq}^{t+1}}] = \mathrm{PCM}\left([\hat{y}_{sq}^t, \hat{y}_{vq}]\right). \tag{5}$$

This process is repeated for four progressive steps in the low-resolution stage. Subsequently, the latent feature is upsampled to the next spatial resolution, and the remaining latents are encoded and decoded in a progressive manner using PCM until the full latent tensor is reconstructed.

By integrating the VQ prior throughout the entire progressive coding pipeline, our method provides consistent global structural guidance for the subsequent coarse-to-fine entropy modeling.

3.3 RD-Oriented Codebook Training Strategy

Effective VQ optimization has long been a well-studied problem in compression (Gersho & Gray, 1992; Feng et al., 2023) and generative models (Mentzer et al., 2023; Fifty et al., 2024; Zhu et al., 2025; Vali et al., 2025). In the most widely used VQ-VAE model (Van Den Oord et al., 2017), the codebook is mainly optimized by matching the selected codeword to the encoder feature:

$$\mathcal{L}_{\mathrm{vq}}^{\mathrm{std}} = \|\mathrm{sg}[z] - e\|_2^2 + \beta\,\|z - \mathrm{sg}[e]\|_2^2\,, \tag{6}$$

where z denotes the encoder feature, e is the selected codeword, β is the commitment weight, and $\mathrm{sg}[\cdot]$ denotes the stop-gradient operator. While this is sufficient for representation learning, it is not fully aligned with the objective of learned image compression. In our framework, the VQ-coded latent $\hat{y}_{vq}$ is not only used for reconstruction, but also participates in entropy modeling. It directly affects both rate and distortion. This makes conventional feature-matching-based codebook optimization (Van Den Oord et al., 2017; Mentzer et al., 2023; Fifty et al., 2024; Zhu et al., 2025; Vali et al.,

Algorithm 1 RD-Oriented Codebook Training Strategy

Input: VQ tokens $\{\mathbf{y}_{vq,t}\}_{t=1}^{M}$, current EMA-based codeword centers $\{\boldsymbol{\mu}_k\}_{k=1}^{K}$, RD-oriented biases $\{\boldsymbol{\delta}_k\}_{k=1}^{K}$, EMA states $\{c_k, \mathbf{s}_k\}_{k=1}^{K}$, decay factor ρ, regularization weight γ, ST factor ω, smoothing constant ϵ

for $t = 1$ to M **do**
  $i_t \leftarrow \arg\min_{k\in\{1,\dots,K\}} \|\mathbf{y}_{vq,t} - \boldsymbol{e}_k\|_2^2$
  $\hat{\mathbf{y}}_{vq,t} \leftarrow \boldsymbol{\mu}_{i_t} + \boldsymbol{\delta}_{i_t}$
end for
for $k = 1$ to K **do**
  $\mathcal{T}_k \leftarrow \{\, t \mid i_t = k \,\}$
  $c_k \leftarrow \rho c_k + (1-\rho)|\mathcal{T}_k|$
  $\mathbf{s}_k \leftarrow \rho \mathbf{s}_k + (1-\rho)\sum_{t\in\mathcal{T}_k} \mathbf{y}_{vq,t}$
  $\boldsymbol{\mu}_k \leftarrow \mathbf{s}_k/(c_k + \epsilon)$
end for
$\mathcal{L}_{\text{aux}} \leftarrow \frac{1}{M}\sum_{t=1}^{M} \left\|\mathbf{y}_{vq,t} - \text{sg}[\boldsymbol{\mu}_{i_t}]\right\|_2^2 + \gamma \cdot \frac{1}{K}\sum_{k=1}^{K} \|\boldsymbol{\delta}_k\|_2^2$
for $t = 1$ to M **do**
  $\bar{\mathbf{y}}_{vq,t} \leftarrow \omega\hat{\mathbf{y}}_{vq,t} + (1-\omega)\,\text{sg}[\hat{\mathbf{y}}_{vq,t}] + (\mathbf{y}_{vq,t} - \text{sg}[\mathbf{y}_{vq,t}])$
end for
Use $\{\bar{\mathbf{y}}_{vq,t}\}_{t=1}^{M}$ for the forward pass and compute $\mathcal{L}_{\text{RD}}$
$\mathcal{L} \leftarrow \mathcal{L}_{\text{RD}} + \mathcal{L}_{\text{aux}}$
Update network parameters and $\{\boldsymbol{\delta}_k\}_{k=1}^{K}$ by back-propagation of $\mathcal{L}$

2025) suboptimal in our setting. To better align codebook learning with the end-to-end compression objective, we further propose an RD-oriented codebook training strategy. The detailed optimization procedure is summarized in Algorithm 1.

Codeword Decomposition. In the proposed strategy, each codeword is decomposed into an exponential moving average (EMA) (Van Den Oord et al., 2017) updated codeword center and an RD-driven offset:

$$\mathbf{e}_k = \boldsymbol{\mu}_k + \boldsymbol{\delta}_k, \tag{7}$$

where $\boldsymbol{\mu}_k$ is updated by EMA without gradient, and $\boldsymbol{\delta}_k$ is optimized by back-propagation from the end-to-end RD loss. In this way, $\boldsymbol{\mu}_k$ provides a stable estimate of the cluster center, while $\boldsymbol{\delta}_k$ allows the codeword to further adapt toward better rate-distortion performance. The detailed EMA update of $\boldsymbol{\mu}_k$ is illustrated in Algorithm 1.

RD-Aware Straight-Through Estimator. To optimize the RD-driven offsets $\boldsymbol{\delta}_k$ with the end-to-end RD loss rather than the feature-matching loss in Eq. 6, we propose an RD-aware straight-through estimator:

$$\bar{\mathbf{y}}_{vq,t} = \omega\hat{\mathbf{y}}_{vq,t} + (1-\omega)\,\text{sg}[\hat{\mathbf{y}}_{vq,t}] + (\mathbf{y}_{vq,t} - \text{sg}[\mathbf{y}_{vq,t}])\,, \tag{8}$$

where ω controls the gradient strength on the codebook. The encoder receives straight-through gradients, while the selected offsets receive ω-scaled gradients from reconstruction distortion and downstream conditional rates. Hard assignments are fixed during back-propagation. Gradients of R_{vq} update the categorical model parameters.

Auxiliary Stabilization Loss. For a more stable codebook update, we use an auxiliary loss with two terms. The first term anchors encoder features to the EMA centers, leaving the offsets to adapt the selected embeddings to downstream RD objectives. The second term is an offset regularization term, which constrains the magnitude of $\boldsymbol{\delta}_k$ and stabilizes its optimization. The auxiliary loss is defined as

$$\mathcal{L}_{\text{aux}} = \frac{1}{M}\sum_{t=1}^{M} \left\|\mathbf{y}_{vq,t} - \text{sg}[\boldsymbol{\mu}_{i_t}]\right\|_2^2 + \gamma \cdot \frac{1}{K}\sum_{k=1}^{K} \|\boldsymbol{\delta}_k\|_2^2, \tag{9}$$

where γ controls the regularization strength on the RD-driven offsets. These constraints are essential to ensure the stable convergence of δ_k within the high-dimensional landscape.

Training Objective. The overall training objective is defined as

$$\mathcal{L} = \mathcal{L}_{\text{RD}} + \mathcal{L}_{\text{aux}}, \tag{10}$$

where the rate-distortion loss $\mathcal{L}_{\text{RD}}$ is given by

$$\mathcal{L}_{\text{RD}} = \mathcal{R}(\hat{y}) + \mathcal{R}(\hat{z}) + \lambda \cdot \mathcal{D}(x, \hat{x}). \tag{11}$$

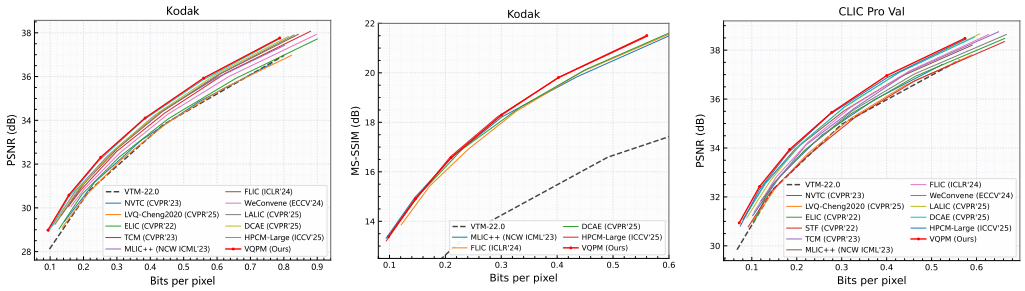


Figure 3: Rate-distortion comparison under PSNR and MS-SSIM on Kodak and CLIC Pro Val.

Here, $\mathcal{R}(\hat{y})$ (including $\mathcal{R}(\hat{y}_{sq})$ and $\mathcal{R}(\hat{y}_{vq})$) and $\mathcal{R}(\hat{z})$ denote the bitrates of $\hat{y}$ and $\hat{z}$, respectively, $\mathcal{D}(x, \hat{x})$ denotes the distortion between x and $\hat{x}$, and λ controls the rate-distortion trade-off.

Overall, the proposed strategy makes codebook training more suitable for learned image compression by combining stable codeword assignment with RD-oriented codeword refinement.

4 Experimental Results

4.1 Experimental Settings

Training settings. We train our models on the Flickr2W dataset (Liu et al., 2020). During training, input images are randomly cropped into 256×256 patches, and the batch size is set to 32. The models are optimized using the loss function defined in Eq. (10). For MSE-optimized models, the Lagrange multiplier λ is selected from $\{0.0018, 0.0035, 0.0067, 0.0130, 0.0250, 0.0483\}$. For MS-SSIM-optimized (Wang et al., 2003) models, λ is selected from $\{2.40, 4.58, 8.73, 16.64, 31.73, 60.50\}$. We use the Adam optimizer (Kingma & Ba, 2015) with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. Training is conducted for 2M steps. The learning rate is initialized as 10^{-4} and reduced to 2×10^{-5}, 5×10^{-6}, and 10^{-6} after 1.6M, 1.8M, and 1.9M steps, respectively.

Evaluation settings. We evaluate our method on Kodak (Kodak), CLIC Professional Validation (Challenge on Learned Image Compression (CLIC), 2020), and Tecnick (Asuni & Giachetti, 2014), which contain 24 images of resolution 512×768, 41 high-quality images, and 100 images of resolution 1200×1200, respectively. We measure bitrate in bits-per-pixel (bpp) from actual arithmetic-coded bitstreams, and distortion using PSNR or MS-SSIM. For clearer RD-curve comparison, MS-SSIM is converted to $-10 \log_{10}(1 - \text{MS-SSIM})$. We also report BD-Rate (Bjontegaard, 2001), using VTM-22.0 (Joint Video Experts Team (JVET), 2023) as the anchor. Encoding and decoding runtimes are measured on an NVIDIA GeForce RTX 3090 GPU. Model complexity, including kMACs/pixel and parameter count, is evaluated using DeepSpeed (Rasley et al., 2020).

Model settings. To constrain VQ complexity, the VQ token dimension and codebook size are set to 32 and 1024, respectively. In RD-oriented codebook training, we use $\omega = 0.3$, $\gamma = 1.0$, EMA decay $\rho = 0.99$, and EMA smoothing constant $\epsilon = 10^{-5}$. Network architectural details are provided in Appendix A. The same VQ configuration is used across all test datasets without data-specific tuning.

4.2 Model Performance

Rate-Distortion Performance. We first compare the proposed VQPM with recent VQ-based learned image compression methods. As illustrated in Fig. 3, compared with LVQ-Cheng2020 (Xu et al., 2025), our method achieves a much better rate-distortion trade-off. Compared with NVTC (Feng et al., 2023), our method also achieves clearly better compression performance. These results verify that VQ can be effectively leveraged for LIC to achieve state-of-the-art rate-distortion performance.

We further compare VQPM with state-of-the-art learned image compression methods (He et al., 2022; Zou et al., 2022; Liu et al., 2023; Jiang & Wang, 2023; Li et al., 2024; Fu et al., 2024; Feng et al., 2025; Lu et al., 2025; Li et al., 2025; Chen et al., 2026a). As shown in Fig. 3, VQPM consistently achieves superior RD performance on Kodak and CLIC Pro Valid under PSNR, and also outperforms recent methods on Kodak under MS-SSIM. Table 1 further reports the BD-Rate results. Compared with VTM-22.0, VQPM reduces PSNR BD-Rate by 20.29%, 19.94%, and 23.95% on Kodak, CLIC Pro Valid, and Tecnick, respectively, achieving state-of-the-art compression performance. Comparisons with practical learned image codecs, including DCVC-RT-Intra (Jia et al., 2025) and JPEG-AI (Esenlik et al., 2026), are provided in Appendix G.

Table 1: Compression performance and complexity comparison. VTM-22.0 serves as an anchor for PSNR BD-Rate calculation. The best compression performance is in **bold**. '–' denotes that the data are unavailable because the code is not open-sourced.

Model	Enc. T (ms)	Dec. T (ms)	kMACs /pixel	Params (M)	BD-Rate Kodak	BD-Rate CLIC Pro Val	BD-Rate Tecnick
NVTC (CVPR'23) (Feng et al., 2023)	33	35	342.97	12.82	-1.01%	-4.20%	-2.58%
LVQ-Cheng2020 (CVPR'25) (Xu et al., 2025)	–	–	956.57	62.69	1.31%	1.14%	–
ELIC (CVPR'22) (He et al., 2022)	45	53	573.88	36.93	-3.22%	-3.89%	-4.57%
TCM (CVPR'23) (Liu et al., 2023)	120	131	1823.58	76.57	-10.70%	-8.32%	-11.84%
MLIC++ (NCW ICML'23) (Jiang & Wang, 2023)	97	99	1282.81	116.72	-15.15%	-14.05%	-17.90%
FLIC (ICLR'24) (Li et al., 2024)	92	94	1096.04	70.96	-13.20%	-9.88%	-15.27%
WeConvene (ECCV'24) (Fu et al., 2024)	153	97	2343.13	107.15	-6.98%	-5.66%	-8.63%
LALIC (CVPR'25) (Feng et al., 2025)	77	79	667.26	66.13	-14.09%	-14.22%	-18.31%
DCAE (CVPR'25) (Lu et al., 2025)	83	82	940.40	119.4	-15.36%	-15.40%	-20.35%
HPCM-Large (ICCV'25) (Li et al., 2025)	91	90	1261.29	89.71	-19.19%	-18.37%	-22.20%
GLIC (CVPR'26) (Chen et al., 2026a)	112	103	1285.55	69.37	-18.38%	-18.07%	-22.70%
VQPM (Ours)	88	85	1083.67	79.42	**-20.29%**	**-19.94%**	**-23.95%**

Table 2: Ablation studies on Vector Quantization-Based Prior Modeling (VQPM) on Kodak.

Method	Prior	CPM	HPM	Subset	BD-Rate	kMACs /pixel	Codebook Usage	Perplexity
Baseline	–	–	–	–	0.00%	1064.81	–	–
SQ-coded prior	SQ	✓	✓	1 group	1.01%	1083.63	–	–
VQCPM only	VQ	✓	–	1 group	-1.29%	1067.04	95.30%	676.34
Full VQPM	VQ	✓	✓	1 group	**-2.08%**	1083.67	**97.95%**	**756.54**
Two-group VQPM	VQ	✓	✓	2 groups	0.35%	1083.83	82.50%	627.34
Three-group VQPM	VQ	✓	✓	3 groups	0.82%	1083.99	75.20%	544.78

Complexity. Table 1 reports neural network inference time, kMACs/pixel, and parameter count. The reported runtime measures only neural network inference and excludes arithmetic coding, since arithmetic coding is implementation-dependent and can vary significantly across released codebases. Compared with HPCM-Large (Li et al., 2025), VQPM obtains larger BD-Rate reduction with lower kMACs/pixel and shorter runtime. The complete latency breakdown is provided in Appendix F.

Visual Comparison. Figure 4 shows visual comparisons of reconstructed images on `kodim03.png` from the Kodak dataset across various learned image compression methods (Li et al., 2025; Lu et al., 2025; Jiang & Wang, 2023). In some texture regions, our method can preserve finer details. More visual comparisons are shown in Appendix I.

4.3 Ablation Studies

Effectiveness of Vector Quantization-Based Prior Modeling. As shown in Table 2, based on the baseline model without VQ-based prior modeling, adding VQCPM reduces BD-Rate by 1.29%, and further introducing VQHPM improves the reduction to 2.08% on the Kodak dataset. This indicates that the two modules provide benefits for both contextual and hyperprior modeling.

Comparison with SQ Prior. To verify that the performance gain comes from the effectiveness of our proposed VQ design rather than only the prior design, we replace the VQ-coded prior with an SQ-coded factorized prior on the same early-coded subset. As shown in Table 2, the SQ-coded prior yields a BD-Rate rise of 1.01%, which is worse than baseline and Full VQPM. This indicates that VQ provides a more effective compact structural prior than scalar quantization for entropy modeling.

VQ-coded subset. We further examine the effect of the VQ-coded subset. By default, VQ is applied to one early-coded subset with channel size $\frac{1}{10}C$. We also evaluate Two-group VQ and Three-group VQ, which extend VQ to the first two and three subsets with channel sizes $\{\frac{1}{10}C, \frac{1}{5}C\}$ and $\{\frac{1}{10}C, \frac{1}{5}C, \frac{3}{10}C\}$, respectively. Here, codebook usage denotes the ratio of activated codewords, and perplexity is computed as $2^{H(p)}$ from the empirical codeword usage distribution p. As shown in Table 2, enlarging the VQ-coded subset consistently degrades RD performance and reduces both codebook usage and perplexity. This indicates that extending VQ to more groups enlarges the VQ optimization space under end-to-end RD training, making codebook learning less tractable. Therefore, a small early-coded subset is sufficient to provide useful global structured prior information while maintaining a manageable optimization space.

Bit Saving Analysis of VQ Prior. Table 3 reports the VQ index rate R_{vq} and rate savings $\Delta R_{\hat{z}}$ and $\Delta R_{\hat{y}_{sq}}$ relative to Baseline on Kodak at $\lambda = 0.013$. The saved/VQ ratio, $(\Delta R_{\hat{z}} + \Delta R_{\hat{y}_{sq}})/R_{vq}$, is 1.84 for one-group VQ and falls below 1 for two and three groups. The same trend holds on Tecnick



Figure 4: Visualization of reconstructed images of `kodim03.png` in the Kodak dataset with various learned image compression methods.

Table 3: VQ prior bit decomposition.

Method	R_{vq}	ΔR_z	$\Delta R_{\hat{y}_{sq}}$	Ratio
1-group	0.0093	0.0045	0.0126	**1.84**
2-group	0.0182	0.0047	0.0132	0.98
3-group	0.0268	0.0051	0.0204	0.95

Table 4: Ablation studies on RD-VQ training on Kodak.

Training Strategy	BD-Rate	Codebook Usage	Perplexity
Standard VQ-VAE	0.00%	78.24%	524.00
Full RD-VQ	**-1.86%**	**97.95%**	**756.54**
W/o RD-aware offset	-1.13%	90.62%	581.32

at low and high rate points as reported in Appendix D.5. When VQ is extended to more groups, the VQ bitrate increases while the saved/VQ ratio drops below 1. These results support restricting VQ to a compact subset whose rate savings exceed its index cost.

RD-Oriented Codebook Training Strategy. Table 4 reports the ablation results of the proposed RD-oriented VQ training strategy (RD-VQ) on Kodak. Full RD-VQ reduces BD-rate by 1.86% relative to standard VQ-VAE (Van Den Oord et al., 2017). Adding the RD-aware offset to the EMA-based variant improves BD-rate by a further 0.73 percentage points and increases codebook usage and perplexity. More hyperparameter ablations are provided in Appendix D.

Backbone Attribution and Transfer. We further evaluate VQPM on the original HPCM-Large backbone. As shown in Appendix D.4, adding VQPM reduces Kodak BD-Rate by 1.99%. This verifies that the RD gain does not depend on the backbone modifications. We also apply VQPM to Channel-AR (Minnen & Singh, 2020), achieving a 1.47% BD-Rate reduction under the same training settings and unchanged VQ hyperparameters. Implementation details and bitrate decomposition are provided in Appendix E. Since channel-wise autoregressive entropy modeling is widely used in LIC, this result provides evidence of the transferability of VQPM. Further designs tailored to Channel-AR may bring additional performance gains.

4.4 Codebook Analysis

We further analyze the learned codebook to understand how the proposed VQ prior behaves in practice. Fig. 5 shows two Kodak images, `kodim13.png` with richer local textures and `kodim20.png` with relatively flatter regions, together with their top-20 most used codewords and counts. For `kodim13.png`, these codewords have similar counts of around 3–6, forming a nearly flat histogram. In contrast, `kodim20.png` exhibits a more concentrated distribution, with several dominant codewords and a long tail of less frequent ones. The most frequently used codeword IDs also differ between the images. These observations indicate content-dependent codebook usage, supporting the role of VQ as a content-adaptive prior. More codebook analysis results are provided in Appendix H.

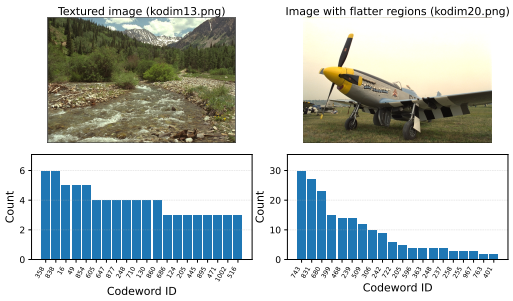


Figure 5: Image-wise codebook usage on Kodak.

5 Conclusion

In this paper, we studied how to effectively exploit vector quantization in learned image compression under a tractable VQ optimization landscape. Instead of using VQ as a full quantizer, we introduced it as a structured prior for entropy modeling and proposed Vector Quantization-Based Prior Modeling (VQPM), which enhances both hyperprior modeling and contextual prior modeling. To further align codebook learning with the end-to-end compression objective, we also proposed an RD-oriented VQ training strategy. Extensive experiments demonstrate that the proposed method achieves state-of-the-art compression performance while maintaining favorable complexity, validating the effectiveness of introducing VQ priors into learned image compression.

AI USE STATEMENT

Generative AI tools were used to assist with literature discovery, experimental table organization, language polishing, sentence-level rewriting, comparison of alternative prose versions, and LaTeX template organization. The authors reviewed the resulting edits, checked the mathematical definitions, experimental settings, numerical values, and citation keys against the source manuscript, and take responsibility for the final content.

REFERENCES

Nicola Asuni and Andrea Giachetti. TESTIMAGES: A large-scale archive for testing visual devices and basic image processing algorithms. In *Smart Tools and Apps for Graphics - Eurographics Italian Chapter Conference*, pp. 63–70, 2014.

Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. In *International Conference on Learning Representations (ICLR)*, 2018.

F. Bellard. BPG Image Format. `http://bellard.org/bpg/`, 2015.

Gisle Bjontegaard. Calculation of average psnr differences between rd-curves. *ITU SG16 Doc. VCEG-M33*, 2001.

Benjamin Bross, Ye-Kui Wang, Yan Ye, Shan Liu, Jianle Chen, Gary J. Sullivan, and Jens-Rainer Ohm. Overview of the versatile video coding (VVC) standard and its applications. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(10):3736–3764, 2021. doi: 10.1109/TCSVT.2021.3101953.

Andrés Buzo, A Gray, RM Gray, and John Markel. Speech coding based upon vector quantization. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(5):562–574, 1980.

Challenge on Learned Image Compression (CLIC). Clic professional validation set, 2020. Dataset used in the Challenge on Learned Image Compression.

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 (CVPR)*, pp. 11315–11325, June 2022.

Jierun Chen, Shiu-hong Kao, Hao He, Weipeng Zhuo, Song Wen, Chul-Ho Lee, and S.-H. Gary Chan. Run, don't walk: Chasing higher FLOPS for faster neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12021–12031, 2023. doi: 10.1109/CVPR52729.2023.01157.

Tong Chen, Haojie Liu, Zhan Ma, Qiu Shen, Xun Cao, and Yao Wang. End-to-end learnt image compression via non-local attention optimization and improved context modeling. *IEEE Transactions on Image Processing*, 30:3179–3191, 2021. doi: 10.1109/TIP.2021.3058615.

Yunuo Chen, Qian Li, Bing He, Donghui Feng, Ronghua Wu, Qi Wang, Li Song, Guo Lu, and Wenjun Zhang. S2cformer: Revisiting the rd-latency trade-off in transformer-based learned image compression. *arXiv preprint arXiv:2502.00700*, 2025. URL `https://arxiv.org/pdf/2502.00700v3`.

Yunuo Chen, Bing He, Zezheng Lyu, Hongwei Hu, Qunshan Gu, Yuan Tian, and Guo Lu. Adaptive learned image compression with graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12150–12161, 2026a.

Yunuo Chen, Zezheng Lyu, Bing He, Hongwei Hu, Qi Wang, Yuan Tian, Li Song, Wenjun Zhang, and Guo Lu. Content-aware mamba for learned image compression. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL `https://openreview.net/forum?id=WwDNiisZQm`.

Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. Learned image compression with discretized Gaussian mixture likelihoods and attention modules. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7936–7945, 2020. doi: 10.1109/CVPR42600.2020.00796.

John Conway and Neil Sloane. Voronoi regions of lattices, second moments of polytopes, and quantization. *IEEE transactions on information theory*, 28(2):211–226, 1982.

Semih Esenlik, Yaojun Wu, Zhaobin Zhang, Ye-Kui Wang, Kai Zhang, Li Zhang, João Ascenso, and Shan Liu. An overview of the JPEG AI learning-based image coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 36(2):2520–2537, 2026. doi: 10.1109/TCSVT.2025.3613244.

Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12873–12883, June 2021.

Donghui Feng, Zhengxue Cheng, Shen Wang, Ronghua Wu, Hongwei Hu, Guo Lu, and Li Song. Linear attention modeling for learned image compression. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pp. 7623–7632, June 2025.

Runsen Feng, Zongyu Guo, Weiping Li, and Zhibo Chen. Nvtc: Nonlinear vector transform coding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6101–6110, 2023.

Christopher Fifty, Ronald G Junkins, Dennis Duan, Aniketh Iyengar, Jerry W Liu, Ehsan Amid, Sebastian Thrun, and Christopher Ré. Restructuring vector quantization with the rotation trick. *arXiv preprint arXiv:2410.06424*, 2024.

Haisheng Fu, Jie Liang, Zhenman Fang, Jingning Han, Feng Liang, and Guohe Zhang. Weconvene: Learned image compression with wavelet-domain convolution and entropy model. In *European Conference on Computer Vision (ECCV)*, pp. 37–53, 2024.

Allen Gersho. Asymptotically optimal block quantization. *IEEE Transactions on information theory*, 25(4): 373–380, 1979.

Allen Gersho and Robert M Gray. *Vector quantization and signal compression*. 1992.

Dailan He, Yaoyan Zheng, Baocheng Sun, Yan Wang, and Hongwei Qin. Checkerboard context model for efficient learned image compression. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14766–14775, 2021. doi: 10.1109/CVPR46437.2021.01453.

Dailan He, Ziming Yang, Weikun Peng, Rui Ma, Hongwei Qin, and Yan Wang. ELIC: Efficient learned image compression with unevenly grouped space-channel contextual adaptive coding. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5708–5717, 2022.

Haiduo Huang, Tian Xia, Wenzhe Zhao, and Pengju Ren. Partial channel network: Compute fewer, perform better. *arXiv preprint arXiv:2502.01303*, 2025.

Herve Jegou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1):117–128, 2010.

Zhaoyang Jia, Jiahao Li, Bin Li, Houqiang Li, and Yan Lu. Generative latent coding for ultra-low bitrate image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26088–26098, 2024.

Zhaoyang Jia, Bin Li, Jiahao Li, Wenxuan Xie, Linfeng Qi, Houqiang Li, and Yan Lu. Towards practical real-time neural video compression. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025, Nashville, TN, USA, June 11-15, 2025*, 2025.

Wei Jiang and Ronggang Wang. MLIC++: Linear complexity multi-reference entropy modeling for learned image compression. In *ICML Workshop*, 2023.

Joint Video Experts Team (JVET). Vvc test model (vtm) 22.0, 2023. Official reference software for Versatile Video Coding (VVC).

Biing-Hwang Juang and A Gray. Multiple stage vector quantization for speech coding. In *ICASSP'82. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 7, pp. 597–600. IEEE, 1982.

Jun-Hyuk Kim, Seungeon Kim, Won-Hee Lee, and Dokwan Oh. Diversify, contextualize, and adapt: Efficient entropy modeling for neural image codec. volume 37, pp. 45956–45974, 2024.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Eastman Kodak. Kodak lossless true color image suite. `http://r0k.us/graphics/kodak/`.

Yuzhuo Kong, Ming Lu, and Zhan Ma. Generative refinement for low bitrate image coding using vector quantized residual. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 14(2):185–197, 2024.

Haowei Kuang, Wenhan Yang, Zongming Guo, and Jiaying Liu. Cross-granularity online optimization with masked compensated information for learned image compression. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16514–16523, 2025.

Han Li, Shaohui Li, Wenrui Dai, Chenglin Li, Junni Zou, and Hongkai Xiong. Frequency-aware transformer for learned image compression. In *International Conference on Learning Representations (ICLR)*, 2024.

Yuqi Li, Haotian Zhang, Li Li, and Dong Liu. Learned image compression with hierarchical progressive context modeling. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025.

Jiaheng Liu, Guo Lu, Zhihao Hu, and Dong Xu. A unified end-to-end framework for efficient deep image compression. *arXiv preprint arXiv:2002.03370*, 2020.

Jinming Liu, Heming Sun, and Jiro Katto. Learned image compression with mixed Transformer-CNN architectures. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14388–14397, 2023. doi: 10.1109/CVPR52729.2023.01383.

Tom D Lookabaugh and Robert M Gray. High-resolution quantization theory and the vector quantizer advantage. *IEEE Transactions on Information Theory*, 35(5):1020–1033, 1989.

Jingbo Lu, Leheng Zhang, Xingyu Zhou, Mu Li, Wen Li, and Shuhang Gu. Learned image compression with dictionary-based entropy model. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pp. 12850–12859, June 2025.

Qi Mao, Tinghan Yang, Yinuo Zhang, Zijian Wang, Meng Wang, Shiqi Wang, Libiao Jin, and Siwei Ma. Extreme image compression using fine-tuned vqgans. In *2024 Data Compression Conference (DCC)*, pp. 203–212. IEEE, 2024.

Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: Vq-vae made simple. *arXiv preprint arXiv:2309.15505*, 2023.

David Minnen and Saurabh Singh. Channel-wise autoregressive entropy models for learned image compression. In *IEEE International Conference on Image Processing (ICIP)*, pp. 3339–3343, 2020. doi: 10.1109/ICIP40778.2020.9190935.

David Minnen, Johannes Ballé, and George Toderici. Joint autoregressive and hierarchical priors for learned image compression. In *Advances in Neural Information Processing Systems*, pp. 10794–10803, 2018.

Yichen Qian, Ming Lin, Xiuyu Sun, Zhiyu Tan, and Rong Jin. Entroformer: A Transformer-based entropy model for learned image compression. In *International Conference on Learning Representations (ICLR)*, 2022.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 3505–3506, 2020.

ML Sabin and R Gray. Product code vector quantizers for waveform and voice coding. *IEEE transactions on acoustics, speech, and signal processing*, 32(3):474–488, 1984.

A. Skodras, C. Christopoulos, and T. Ebrahimi. The JPEG 2000 still image compression standard. *IEEE Signal Processing Magazine*, 18(5):36–58, 2001. doi: 10.1109/79.952804.

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.

Mohammad Hassan Vali, Tom Bäckström, and Arno Solin. Diveq: Differentiable vector quantization using the reparameterization trick. *arXiv preprint arXiv:2509.26469*, 2025.

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

Gregory K Wallace. The JPEG still picture compression standard. *Communications of the ACM*, 34(4):30–44, 1991.

Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pp. 1398–1402. Ieee, 2003.

Hao Xu, Xiaolin Wu, and Xi Zhang. Multirate neural image compression with adaptive lattice vector quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7633–7642, 2025.

Naifu Xue, Qi Mao, Zijian Wang, Yuan Zhang, and Siwei Ma. Unifying generation and compression: Ultra-low bitrate image coding via multi-stage transformer. In *2024 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6. IEEE, 2024.

Lijun Yu, Yong Cheng, Kihyuk Sohn, José Lezama, Han Zhang, Huiwen Chang, Alexander G. Hauptmann, Ming-Hsuan Yang, Yuan Hao, Irfan Essa, and Lu Jiang. Magvit: Masked generative video transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10459–10469, June 2023.

Haotian Zhang, Li Li, and Dong Liu. Generalized Gaussian model for learned image compression. *arXiv preprint arXiv:2411.19320*, 2024.

Haotian Zhang, Yuqi Li, Li Li, and Dong Liu. Learning switchable priors for neural image compression. *IEEE Transactions on Circuits and Systems for Video Technology*, 2025.

Xi Zhang and Xiaolin Wu. Lvqac: Lattice vector quantization coupled with spatially adaptive companding for efficient learned image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10239–10248, 2023.

Xi Zhang and Xiaolin Wu. Learning optimal lattice vector quantizers for end-to-end neural image compression. In *Advances in Neural Information Processing Systems*, volume 37, pp. 106497–106518, 2024.

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*, pp. 22968–22977, 2025.

Renjie Zou, Chunfeng Song, and Zhaoxiang Zhang. The devil is in the details: Window-based attention for image compression. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17471–17480, 2022. doi: 10.1109/CVPR52688.2022.01697.

The appendix presents detailed network architectures in Sec. A and the channel partition design of VQPM in Sec. B. Additional evaluations include rate–distortion results in Sec. C, ablation studies in Sec. D, cross-architecture evaluation in Sec. E, complete codec latency measurements in Sec. F, and comparisons with practical learned image codecs in Sec. G. Further codebook analysis and visual comparisons are provided in Sec. H and Sec. I, respectively.

A Network Structures

Figure 7 shows the detailed architectures of the main network components in Fig. 1(b) of the main paper, including four transform networks, i.e., g_a, g_s, h_a, and h_s, as well as the entropy parameter network g_{ep} used in the hierarchical context model. The transform networks are built by strided convolution/deconvolution layers and lightweight nonlinear blocks. The entropy parameter network g_{ep} is built by convolution blocks with step-adaptive modulation.

Transform networks. As shown in Fig. 7 (a)–(d), the four transform networks adopt a multi-stage design. The analysis transform g_a maps the input image to the main latent y through four resolution stages, with $L1$ and $L2$ PCBlocks (Chen et al., 2023) used at the first two stages and $L3$ ATPCBlocks (Huang et al., 2025) used at the low-resolution stage. The synthesis transform g_s approximately mirrors g_a and reconstructs the image from $\hat{y}$ through four upsampling stages. The first downsampling layer and the last upsampling layer use 4×4 kernels, while all other resolution changes are implemented by 2×2 convolution/deconvolution layers. The detailed architectures of PCBlock and ATPCBlock are illustrated in Fig. 6 (a) and (b). The hyper-analysis transform h_a converts y to the hyper latent z through two downsampling stages with $L4$ and $L5$ PCBlocks, while the hyper-synthesis transform h_s decodes $\hat{z}$ through two corresponding upsampling stages to generate hyperprior features for subsequent entropy modeling.

Entropy parameter networks. In the hierarchical context model (Li et al., 2025), we further use the entropy parameter network g_{ep} to predict the parameters of the latent features from the concatenated prior features. As shown in Fig. 7 (e), g_{ep} is built by a stack of DepthConvBlocks with a step-adaptive modulation module. The detailed architecture of DepthConvBlock is illustrated in Fig. 6 (c). Given the input feature, g_{ep} first applies $N1$ DepthConvBlocks for feature transformation, then applies a step-dependent embedding, and finally uses another $N2$ DepthConvBlocks to generate the output parameters. To reduce model complexity, the main body of g_{ep} is shared across different coding steps, while lightweight step-specific adaptors are used to preserve step flexibility. The final 1×1 convolution maps the transformed feature to $2C$ channels, corresponding to the predicted means and scales.

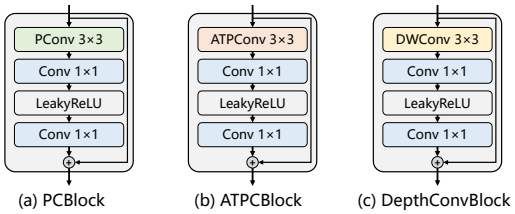


Figure 6: Network structures of (a) PCBlock, (b) ATPCBlock, and (c) DepthConvBlock.

Partial attention convolution. The ATPCBlocks used in the analysis and synthesis transforms adopt an adapted `PAT_sf` variant of PATConv (Huang et al., 2025). Specifically, the ATPConv operation splits the input feature into two channel subsets, applies a 3×3 convolution and window-based self-attention with relative position encoding to the two subsets, respectively, and concatenates their outputs. The convolution and attention branches process $1/4$ and $3/4$ of the input channels, respectively. Each ATPCBlock consists of the ATPConv operation, a 1×1 convolution, LeakyReLU, and another 1×1 convolution, with a residual connection around the block. These blocks are used after the third downsampling stage of the analysis transform and at the corresponding stages of the synthesis transform. The effect of the backbone modifications is evaluated separately from VQPM in Sec. D.4.

Block configuration. In the implementation, the stage depths of the four transform networks are set to $[L1, L2, L3, L4, L5] = [2, 2, 8, 1, 3]$. The numbers of DepthConvBlocks are set as $[N1, N2] = [2, 2]$ and $[N1, N2] = [3, 2]$ in g_{ep} for low-resolution features and high-resolution features, respectively.

B Channel Partition Design

In VQPM, we divide the latent $y \in \mathbb{R}^{C \times \frac{H}{16} \times \frac{W}{16}}$ into eight channel groups evenly along the channel dimension, where each group has size $\frac{C}{8} \times \frac{H}{16} \times \frac{W}{16}$.

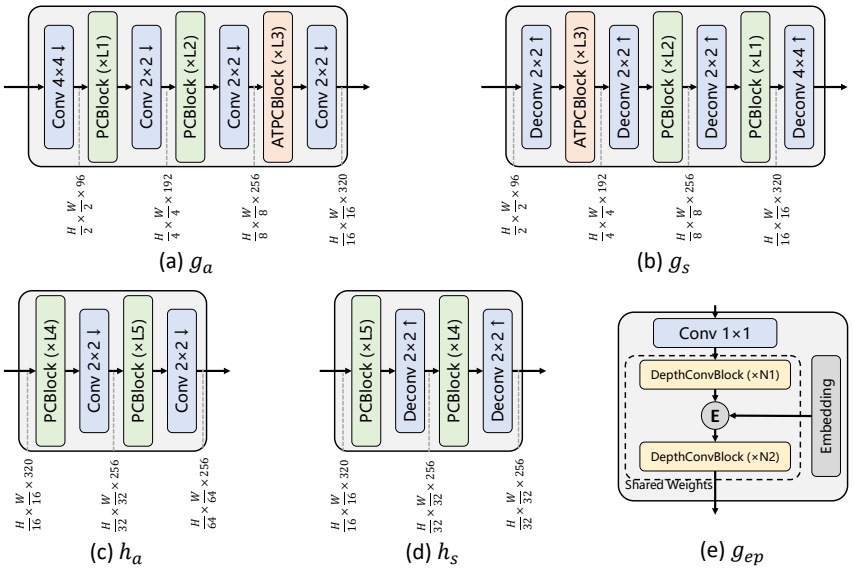


Figure 7: Network structures of (a) the analysis transform g_a; (b) the synthesis transform g_s; (c) the hyper-analysis transform h_a; (d) the hyper-synthesis transform h_s; and (e) the entropy parameter network g_{ep}.

Table 5: Ablation studies on codebook configurations. The best results are in **bold**.

Codebook Size	32	128	1024	2048
RD Loss	0.7404	0.7388	**0.7359**	0.7399

ω	0.0	0.01	0.03	0.1	0.3	0.8	1.0
RD Loss	0.7459	0.7374	0.7373	0.7367	**0.7359**	0.7370	0.7372

γ	0.001	0.01	0.03	0.1	0.3	1	3
RD Loss	0.7370	0.7373	0.7374	0.7364	0.7367	**0.7359**	0.7363

For each group, one spatial position is selected from every 2×2 spatial block to form a down-scaled feature of size $\frac{C}{8} \times \frac{H}{32} \times \frac{W}{32}$. As shown in Fig. 8, four complementary interleaved sampling patterns are used for Channel Groups 1–4 and repeated for Channel Groups 5–8. The extracted features are concatenated along the channel dimension to obtain the down-scaled latent $\in \mathbb{R}^{C \times \frac{H}{32} \times \frac{W}{32}}$, which is subsequently partitioned for the VQ-based prior modeling.

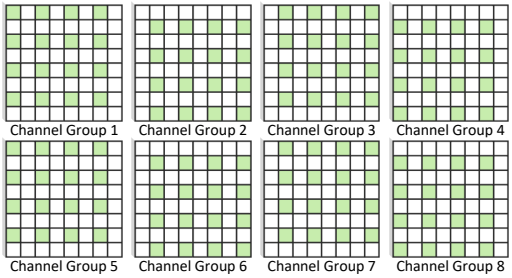


Figure 8: Illustration of the channel partition design in VQPM.

C Additional Rate-Distortion Performance

Figure 9 presents the rate-distortion performance of our model optimized for MSE on the Tecnick dataset. Figure 10 presents the rate-distortion performance of our model optimized for MS-SSIM on the CLIC Pro Valid and the Tecnick datasets. Compared to recent advanced learned image compression methods (Jiang & Wang, 2023; Li et al., 2024; Lu et al., 2025; Li et al., 2025), our proposed method achieves superior performance.

D Additional Ablation Studies

We provide additional ablation studies on the codebook configurations, including the codebook size, codebook gradient factor ω, and regularization weight γ. Similar trends are observed at other rate points. When ablating one factor, the other two are fixed to their default settings, i.e., the codebook size is 1024, $\omega = 0.3$, and $\gamma = 1$.

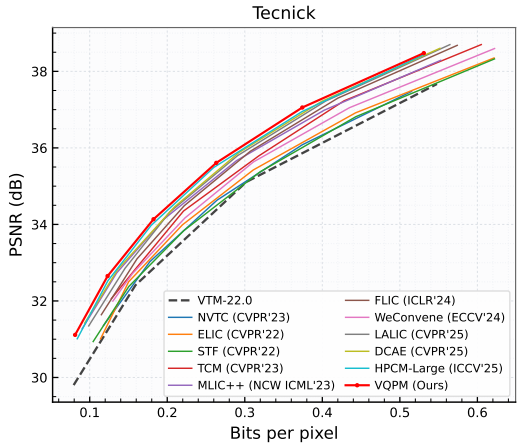


Figure 9: Additional rate-distortion curves in terms of PSNR on the Tecnick dataset.

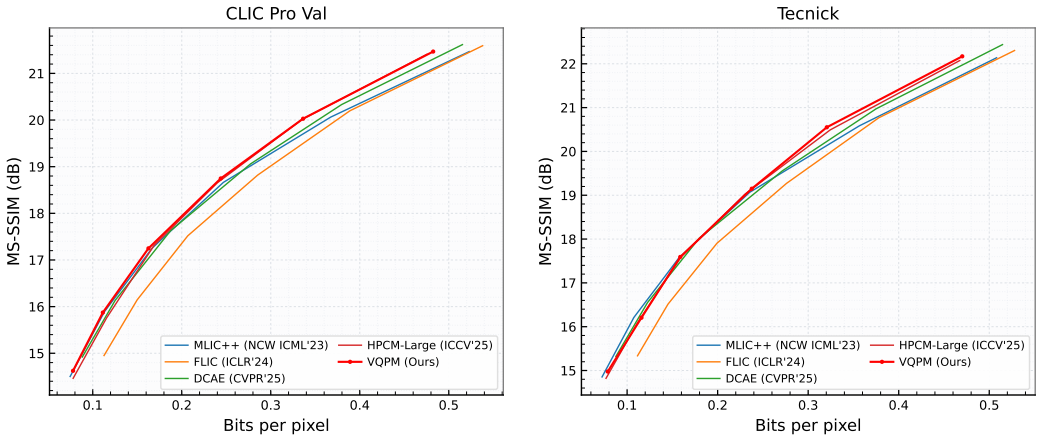


Figure 10: Additional rate-distortion curves in terms of MS-SSIM on the CLIC Pro Valid dataset (left) and the Tecnick dataset (right).

D.1 Ablation Studies on Codebook Sizes

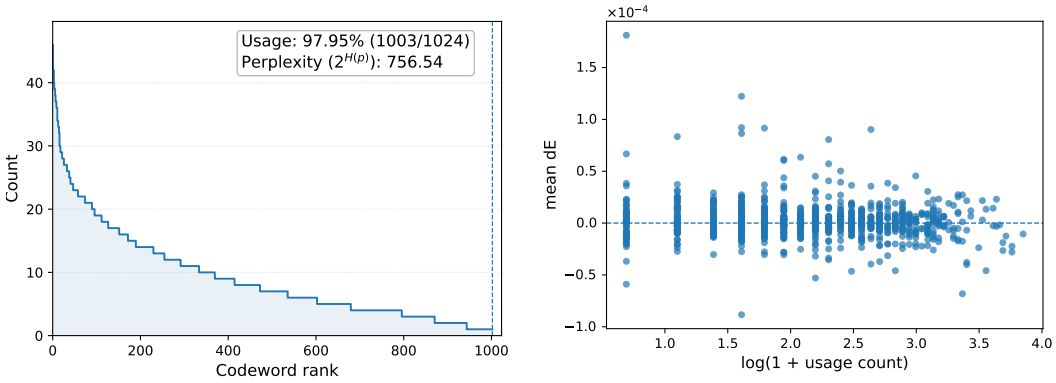
The first line of Table 5 presents the effect of different codebook sizes. Experiments are conducted using $\lambda = 0.013$. As the codebook size increases from 32 to 1024, the RD loss consistently decreases, indicating that a larger codebook provides stronger representation capacity for the VQ prior. However, further increasing the size to 2048 leads to a slight performance drop. This suggests that a larger codebook may increase the VQ complexity and bit length while providing limited additional benefit to the prior modeling process, and thus may not improve the overall rate-distortion performance. Therefore, we use the codebook size 1024 in our final model.

D.2 Ablation Studies on Codebook Gradient Factor ω

The second line of Table 5 shows the influence of the codebook gradient factor ω. Experiments are conducted using $\lambda = 0.013$. When $\omega = 0$, the RD-oriented offset δ_k in the codeword does not receive the RD gradient, leading to inferior performance. As ω increases from 0 to 0.3, the RD loss gradually improves, which verifies the effectiveness of introducing RD-driven codebook updates. When ω becomes too large, the performance degrades. Based on this result, we set $\omega = 0.3$ in the final model.

D.3 Ablation Studies on Regularization Weight γ

The third line of Table 5 reports the effect of the regularization weight γ. Experiments are conducted using $\lambda = 0.013$. Very small or very large regularization weights both lead to worse performance, while a moderate value gives the best result. This result suggests that a suitable regularization term is helpful for stabilizing codebook learning. Accordingly, we use $\gamma = 1$ in our final experiments.



(a) Global codeword usage rank.

(b) Mean dE versus usage frequency.

Figure 11: Codebook usage and distortion analysis. Left: global usage distribution of learned codewords. Right: relationship between codeword usage frequency and the corresponding mean dE, where each point denotes one codeword and the x-axis is $\log(1 + \text{usage count})$.

Table 6: Ablation studies on backbone modifications and VQPM on Kodak. HPCM-Large without VQPM serves as the BD-Rate anchor.

Backbone	VQPM	BD-Rate	kMACs/pixel	Params (M)
HPCM-Large	–	0.00%	1261.29	89.71
HPCM-Large	✓	**-1.99%**	1280.27	89.82
Current efficient backbone	–	+0.99%	1064.81	79.31
Current efficient backbone	✓	-1.18%	1083.67	79.42

D.4 Ablation Studies on Backbone Modifications

We evaluate VQPM on both the original HPCM-Large backbone and the current efficient backbone to separate the contribution of VQ-based prior modeling from the backbone modifications. Table 6 reports the results on Kodak, where all BD-Rate values use HPCM-Large without VQPM as the anchor. On the original HPCM-Large backbone, adding VQPM reduces BD-Rate by 1.99%, with an increase of 18.98 kMACs/pixel and 0.11M parameters. This result verifies that VQPM is effective without the PATConv-based transform modification. The current efficient backbone reduces computation and parameter count, while its BD-Rate increases by 0.99% without VQPM. Adding VQPM to this backbone improves the result to -1.18% relative to HPCM-Large. These results show that VQPM provides the rate-distortion gain, while the backbone modifications account for most of the complexity reduction. Using the current efficient backbone without VQPM as the anchor, the direct same-backbone comparison gives the 2.08% BD-Rate reduction reported in Table 2.

D.5 Additional Bit Saving Analysis

We extend the bit saving analysis in Table 3 to the 100-image Tecnick dataset at a low-rate point ($\lambda = 0.0018$) and a high-rate point ($\lambda = 0.0483$). Table 7 reports the VQ index bitrate R_{vq} and the saved bits of $\hat{z}$ and $\hat{y}_{sq}$ against the baseline without VQPM. All bitrate values are measured in bpp, and the saved/VQ ratio is computed as $(\Delta R_{\hat{z}} + \Delta R_{\hat{y}_{sq}})/R_{vq}$. The one-group setting achieves ratios of 1.61 and 2.07 at the two rate points, respectively, indicating that the subsequent bitrate savings exceed the cost of transmitting the VQ indices. For two-group and three-group VQ, the ratios remain below 1 at both rate points. Together with the Kodak result, these observations show that a compact VQ subset provides a better balance between index cost and subsequent bitrate savings than extending VQ to more groups.

E Cross-Architecture Evaluation

Integration into Channel-AR. We further evaluate VQPM on Channel-AR (Minnen & Singh, 2020), which uses channel-wise autoregressive entropy modeling. We divide the main latent into 10 channel groups and apply VQ to the first spatial position of every non-overlapping 2×2 block within the first group. This produces VQ tokens of size $\frac{C}{10} \times \frac{H}{32} \times \frac{W}{32}$, while the remaining latents retain scalar quantization. The VQ tokens are decoded first and are then used by VQHPM and VQCPM to model the hyperlatent and the remaining scalar-quantized

Table 7: VQ prior bit decomposition on Tecnick at low and high rate points. Bitrates are measured in bpp.

Rate point	VQ subset	R_{vq}	$\Delta R_{\hat{z}}$	$\Delta R_{\hat{y}_{sq}}$	Saved/VQ
$\lambda = 0.0018$	1 group	0.0084	0.0022	0.0113	**1.61**
	2 groups	0.0162	0.0019	0.0121	0.86
	3 groups	0.0243	0.0022	0.0147	0.70
$\lambda = 0.0483$	1 group	0.0099	0.0056	0.0149	**2.07**
	2 groups	0.0198	0.0046	0.0151	0.99
	3 groups	0.0292	0.0066	0.0184	0.86

Table 8: Cross-architecture evaluation on Kodak. Channel-AR serves as the BD-Rate anchor.

Method	BD-Rate	Params (M)	kMACs/pixel
Channel-AR	0.00%	39.90	292.81
Channel-AR + VQPM	**-1.47%**	40.03	304.32

latents, respectively. This design preserves the compact, early-decoded VQ prior and its use in subsequent entropy modeling.

Experimental settings. Channel-AR and Channel-AR with VQPM use the same training data, rate points, optimization settings, and training schedule. We retain the VQ hyperparameters and RD-oriented codebook training strategy used in the HPCM experiments without architecture-specific tuning. Specifically, we use a codebook size of 1024, VQ token dimension of 32, $\omega = 0.3$, $\gamma = 1$, EMA decay $\rho = 0.99$, and smoothing constant $\epsilon = 10^{-5}$. The results are evaluated on Kodak.

Rate-distortion performance. As shown in Table 8, VQPM reduces BD-Rate by 1.47% relative to Channel-AR, with parameter count increasing from 39.90M to 40.03M and computation increasing from 292.81 to 304.32 kMACs/pixel. This result demonstrates that the VQ prior also improves rate-distortion performance in a channel-autoregressive entropy model.

Bit saving analysis. Table 9 further decomposes the VQ index cost and the subsequent bitrate savings relative to Channel-AR. At the low and high rate points, the saved/VQ ratios are 1.33 and 1.48, respectively. In both cases, the savings in the hyperlatent and remaining scalar-quantized latents exceed the VQ index cost, consistent with the observations on the HPCM-based model.

F Complete Codec Latency

The BD-Rate results in Table 1 are computed using the sizes of actual arithmetic-coded bitstreams. The runtime in that table measures neural network inference only. Table 10 additionally reports entropy-coding time and complete encoding and decoding time under the same hardware and evaluation protocol, using an NVIDIA GeForce RTX 3090 GPU for neural network inference. The reported totals include arithmetic coding and decoding, bitstream packing, and bitstream I/O.

VQPM requires 115 ms for encoding and 113 ms for decoding, compared with 119 ms and 122 ms for HPCM-Large, respectively. VQ-index entropy encoding and decoding each take approximately 3 ms, since VQ is applied to only a small low-resolution subset. Thus, VQPM retains a modest latency advantage over HPCM-Large after entropy coding is included. Entropy-coding time also depends on the probability factorization, sequential coding schedule, and arithmetic-coder implementation. We therefore report these totals as practical coding latency, alongside kMACs/pixel and neural network inference time for architectural complexity comparison.

G Comparison with Practical Learned Image Compression Methods

Experimental settings. We further compare VQPM with DCVC-RT-Intra (Jia et al., 2025) and JPEG-AI (Esenlik et al., 2026) on Kodak. JPEG-AI is evaluated at the high operating point (HOP) with coding tools enabled. Table 11 reports PSNR BD-Rate relative to VTM-22.0, computational complexity, and complete encoding and decoding times on an NVIDIA GeForce RTX 3090, including entropy coding.

Table 9: VQ prior bit decomposition for Channel-AR on Kodak. Bitrates are measured in bpp.

Rate point	R_{vq}	$\Delta R_{\hat{z}}$	$\Delta R_{\hat{y}_{sq}}$	Saved/VQ
Low ($\lambda = 0.0018$)	0.0081	0.0019	0.0089	**1.33**
High ($\lambda = 0.0483$)	0.0088	0.0024	0.0106	**1.48**

Table 10: Complete codec latency. All times are measured in milliseconds. For VQPM, the parenthesized values denote the VQ-index portion already included in the entropy-coding time.

Method	NN enc.	Entropy enc.	Total enc.	NN dec.	Entropy dec.	Total dec.
ELIC	45	78	123	53	67	120
TCM	120	108	228	131	120	251
MLIC++	97	101	198	99	125	224
HPCM-Large	91	28	119	90	32	122
VQPM (Ours)	88	27 (3)	115	85	28 (3)	113

Rate-distortion performance and complexity. VQPM achieves the best PSNR RD performance among the compared methods, with a BD-Rate reduction of 20.29% over VTM-22.0, compared with 10.22% for DCVC-RT-Intra and a BD-Rate increase of 26.32% for JPEG-AI. DCVC-RT-Intra and JPEG-AI require fewer kMACs/pixel and achieve lower encoding and decoding latency. Both methods are engineered for practical LIC deployment, with optimizations that contribute to their shorter encoding and decoding times. VQPM prioritizes advancing state-of-the-art RD performance, targeting a different rate–distortion–complexity trade-off with larger bitrate savings under the PSNR criterion.

JPEG-AI under perceptual quality metrics. JPEG-AI uses a weighted combination of MSE and perceptual metrics during training and evaluates coding performance with perceptual quality metrics (Esenlik et al., 2026). Table 12 presents its published Kodak results under the official evaluation setting. JPEG-AI achieves BD-Rate reductions of 18.4% across the metrics. These results highlight JPEG-AI's strong perceptual performance, while VQPM is optimized for MSE and evaluated primarily in terms of PSNR.

H Additional Codebook Analysis

H.1 Global Usage Statistics

As shown in Fig. 11a, the learned codebook exhibits broad global coverage without collapse. Across Kodak, 1003 out of 1024 codewords are activated, indicating that the codebook is widely utilized rather than dominated by a small subset of codewords. Moreover, the global perplexity is 756.54, computed as $2^{H(p)}$ from the normalized codeword usage distribution p, which indicates that a large number of codewords are effectively contributing to representation rather than remaining rarely used.

H.2 Behavior of the RD-Oriented Offset

To better understand the effect of the RD-oriented offset, we analyze

$$dE = \|z - (\mu_k + \delta_k)\|_2^2 - \|z - \mu_k\|_2^2,$$

where z denotes the assigned latent token, μ_k is the EMA codeword center, and δ_k is the learned RD-oriented offset of the selected codeword. A negative dE indicates that the RD-oriented offset moves the selected codeword closer to the assigned latent token in Euclidean space, whereas a positive dE indicates the opposite.

Figure 11b plots the mean dE of each codeword against its usage frequency. The usage count and mean dE of each codeword are computed on the Kodak dataset at the $\lambda = 0.013$ rate point. For infrequently used codewords, the mean dE values are more dispersed. They can also reach relatively large magnitudes. By contrast, the mean dE values of frequently used codewords are much more concentrated. This suggests that rarely used codewords tend to have more variable offset behavior, while frequently used codewords exhibit more consistent refinement patterns.

H.3 Additional Image-wise Codebook Behavior

Figure 12 extends the image-wise codebook analysis in the main paper to all 24 Kodak images. For each image, we show the original content together with the histogram of its top-20 most frequently used codewords.

Table 11: Comparison with practical learned image codecs on Kodak. Encoding and decoding times include entropy coding and are measured on an NVIDIA GeForce RTX 3090. BD-Rate is evaluated using PSNR with VTM-22.0 as the anchor.

Method	Enc. time (ms)	Dec. time (ms)	kMACs /pixel	Kodak BD-Rate
DCVC-RT-Intra	25	23	532.19	-10.22%
JPEG-AI (HOP + tools on)	43	33	205.22	+26.32%
VQPM (Ours)	115	113	1083.67	**-20.29%**

Table 12: JPEG-AI perceptual performance on Kodak under the official evaluation setting. Results are taken from Esenlik et al. (2026). All entries are BD-Rates (%). AVG denotes the reported average across the six metrics.

VIF	FSIM	NLPD	IW-SSIM	VMAF	PSNR-HVS	AVG
0.0	-28.8	-15.6	-32.0	-38.3	+4.4	-18.4

Consistent with the observations in the main paper, the codeword usage behavior varies noticeably across images. Some images show relatively flat histograms, suggesting more diverse codeword usage, while others show more concentrated histograms, where a few codewords are reused more frequently. Meanwhile, the identities of the most frequently used codewords also vary substantially across images. These results further support that the learned codebook, although globally shared, is utilized in a strongly content-dependent manner.

I Additional Visual Comparison Results

Figure 13 and Fig. 14 present the reconstructed images of our proposed VQPM and several recent methods (He et al., 2022; Jiang & Wang, 2023; Lu et al., 2025; Li et al., 2025).



Figure 12: Additional image-wise codebook behavior on all 24 Kodak images. For each image, we show the original content together with the histogram of its top-20 most frequently used codewords.



Figure 13: Visual comparison on `kodim09.png` from the Kodak dataset. We show the original image and the reconstructed results of different learned image compression methods, together with their bitrates (bpp) and PSNR values.

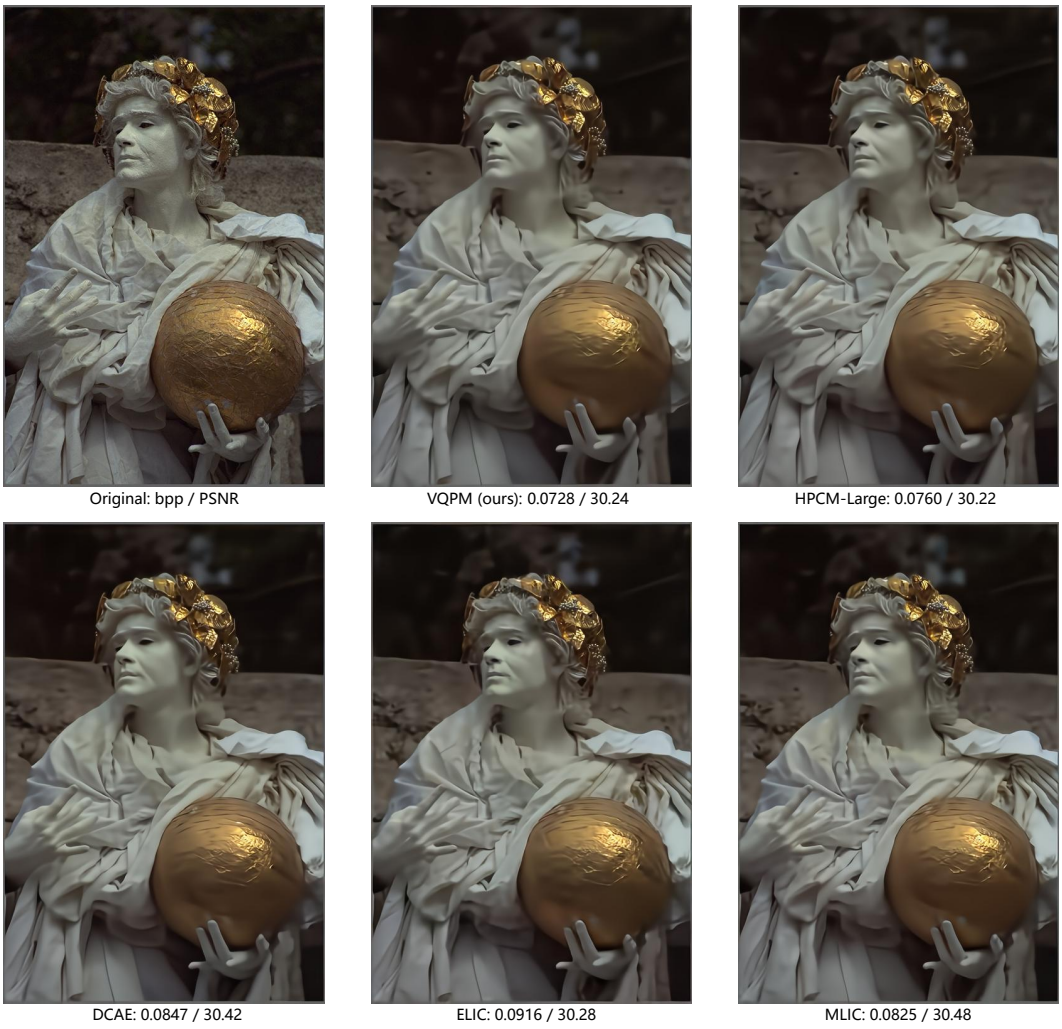


Figure 14: Visual comparison on `kodim17.png` from the Kodak dataset. We show the original image and the reconstructed results of different learned image compression methods, together with their bit rates (bpp) and PSNR values.