Learned Image Compression Framework with Quad-Prior Entropy Model

Feifeng Wang
Shanghai University
Shanghai, China
wff0520@shu.edu.cn

Yao zhu
Shanghai University
Shanghai, China
yaozhu@shu.edu.cn

Zhaoyi Tian
Shanghai University
Shanghai, China
kinda@shu.edu.cn

Liquan Shen
Shanghai University
Shanghai, China
jsslq@163.com

***Abstract*—Learned image compression has emerged as a promising alternative to traditional codec standards, achieving superior rate-distortion performance by leveraging deep neural networks. However, balancing computational efficiency and compression performance remains a critical challenge in entropy modeling. While autoregressive models capture rich context, they suffer from slow sequential decoding. Parallel models improve speed but underutilize spatial and channel dependencies. To address this trade-off, we propose a learned image compression framework with quad-prior entropy model based on a quadtree-inspired partitioning strategy. Our method divides the latent representation into four groups along the channel dimension and partitions each group into non-overlapping 2×2 spatial patterns. Entropy coding proceeds in four sequential steps, where each step encodes one position within the block using progressively accumulated context from previous steps. This design enables the model to utilize up to 8 spatial neighbors on average—twice that of prior parallel models—and exploits cross-channel correlations through inter-group context sharing. Moreover, all positions within each step are encoded in parallel, ensuring high computational efficiency. When integrated into an end-to-end compression framework with a main autoencoder network and quantization parameter (QP) embedding for variable bitrate control, the proposed method achieves state-of-the-art performance on benchmark datasets.**

***Index Terms*—Learned image compression, entropy modeling, variable bitrate.**

I. Introduction

Image compression is a fundamental technology in digital media systems that enables the efficient storage and transmission of visual data. Traditional codecs such as JPEG [1], HEVC [2], and VVC [3] rely on handcrafted transforms and entropy coding schemes. In contrast, learned image compression leverages deep neural networks to jointly optimize the entire pipeline—from feature extraction to entropy modeling—resulting in improved rate-distortion (R-D) performance.

A key component in learned compression is the entropy model, which estimates the probability distribution of quantized latent variables for arithmetic coding [4]. Spatial autoregressive models achieve high accuracy by conditioning each symbol on its spatial predecessors, but suffer from slow sequential decoding [5], [6]. To improve efficiency, parallel models [7] such as, channel-wise autogressive [8], checkerboard [9] and dual Spatial entropy model [10] have been proposed. However, these methods use limited context, leading to suboptimal compression.

In this paper, we present Quad-Prior, a practical learned image compression framework that integrates a fine-grained contextual entropy model with a QP-aware main codec network. Inspired by quadtree partitioning, Quad-Prior introduces a four-step coding scheme that divides the latent space into four channel groups and processes 2×2 spatial patterns in a structured order. This design progressively accumulates spatial and channel context, significantly enhancing entropy modeling accuracy while maintaining intra-step parallelism for efficient coding. We further embed the quantization parameter (QP) directly into both the encoder and decoder, enabling flexible variable bitrate (VBR) control at inference time with a single model. Experimental results demonstrate state-of-the-art rate-distortion performance on standard benchmarks, outperforming existing methods in both objective metrics (PSNR, MS-SSIM) and perceptual quality.

II. Methodology

A. Overview

The proposed framework consists of two main components: (1) a main coding network for feature transformation and reconstruction, and (2) a Quad-Prior contextual entropy model for efficient entropy coding. The overall architecture follows an encoder-decoder structure with side hyperpriors for entropy modeling, as shown in Fig. 1.

B. Main Coding Network

Encoder. Given input image x$\in \mathbb{R}^{H\times W\times 3}$, the encoder first applies a Patchify 8↓ operation to downsample the spatial resolution by 8× and extract local patches. The resulting feature map passes through a Depthwise Convolution Block (DCBlock) for initial feature extraction. The DCBlock consists of two stacked residual blocks, each incorporating depthwise convolutions [5] instead of vanilla convolutions; these depthwise convolutions reduce computational complexity while preserving local spatial information. A quantization parameter Q_e is then embedded via element-wise multiplication:

$$y_e = y \cdot Q_e \tag{1}$$

enabling variable bitrate control. Six stacked DCBlock(M, M) layers refine the features, followed by Conv(M, N) 2↓ for final downsampling. The output y is scalar-quantized to produce discrete code $\hat{y}$.

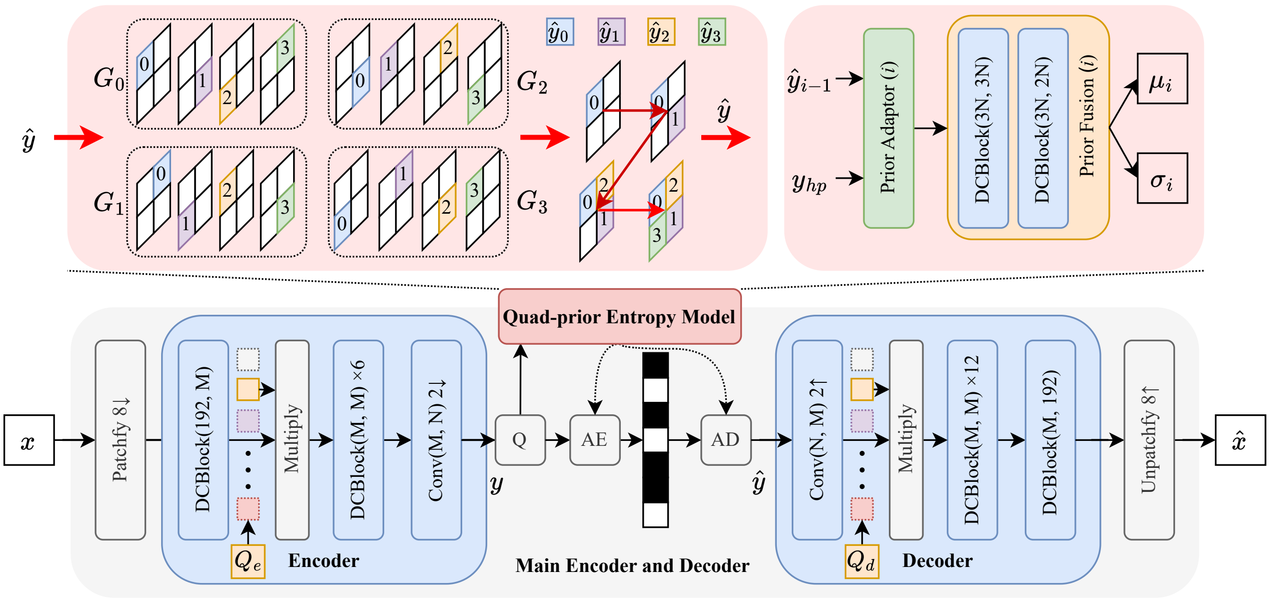

Fig. 1. The overall flowchart of the proposed image compression network.

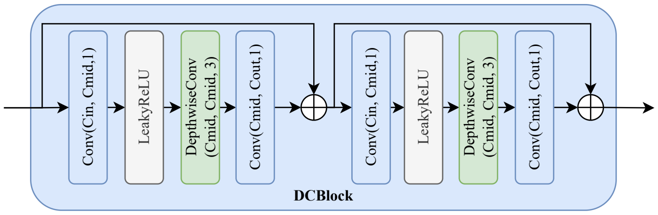

Fig. 2. The structure of Depthwise Convolution Block (DCBlock).

Decoder. The decoder reverses the process: Conv(N, M) 2↑ upsamples the latent code, followed by multiplication with Q_d.

$$\hat{y_d} = \hat{y} \cdot Q_d \tag{2}$$

Twelve DCBlock(M, M) layers recover fine details, then DCBlock(M, 192) maps to patch-level features. Finally, Unpatchify 8↑ reconstructs the full-resolution image $\hat{x}$.

C. Quad-Prior Entropy Model

Latent Space Partitioning. To achieve high compression efficiency, accurate entropy modeling of the quantized latent representation $\hat{y} \in \mathbb{R}^{h \times w \times N}$is essential. In quad-prior entropy model, $\hat{y}$ is divided into four groups G_0, G_1, G_2, G_3, along the channel dimension (N/4 channels per group). Within each group, the spatial map is divided into non-overlapping 2 × 2 blocks, including $\hat{y}_0$, $\hat{y}_1$, $\hat{y}_2$ and $\hat{y}_3$, which form the basis for fine-grained entropy coding. The core idea of Quad-Prior is to encode these blocks in four sequential steps, where each step processes one relative position (e.g., top-left, bottom-right, top-right, bottom-left) across all spatial locations and channel groups.

Structured Encoding/Decoding Order. In Step 0, the top-left positions are encoded using only a global hyperprior as context. In Step 1, the top-right positions are predicted conditioned on the already-decoded top-left symbols within the same block. Step 2 encodes bottom-left positions using both previously decoded horizontal neighbors, and finally, in Step 3, the bottom-right positions are modeled with full access to the other three positions in the block—enabling up to 8 contextual neighbors (including spatial and cross-channel dependencies). This hierarchical accumulation of context significantly increases the average number of available neighbors to 4 per symbol, doubling that of conventional parallel models like Checkerboard or Dual Spatial, and matching the context density of autoregressive approaches. Crucially, all positions within each step are encoded in parallel, ensuring high throughput and decoding speed. Furthermore, our model exploits cross-group dependencies: during Step 3, channels from other groups that have been partially decoded serve as additional inter-channel context, enhancing probability estimation accuracy.

For each step *i*, a dedicated Prior Adaptor network takes the hyperprior features y_{hp} and the already-decoded symbols $\hat{y}_{<i}$ to predict location-adaptive gaussian parameters, like means μ_i and scales σ_i, which are then fused with the current latent features in a Prior Fusion module to form the final conditional distribution $p(\hat{y}_i|\hat{y}_{<i}, y_{hp})$. This dynamic, context-aware prior generation enables precise entropy coding and minimizes bitrate. Overall, Quad-Prior achieves an optimal trade-off between modeling power and efficiency, making it highly effective for learned image compression.

D. Loss Functions

For the **objective quality stage**, similar to most existing learned image compression methods, we optimize the model by minimizing the joint rate-distortion trade-off. The loss function is formulated as:

$$\mathcal{L} = R(\hat{\boldsymbol{y}}) + R(\hat{\boldsymbol{z}}) + \lambda \cdot D(\boldsymbol{x}, \hat{\boldsymbol{x}}), \tag{3}$$

where $R(\hat{\boldsymbol{y}})$ and $R(\hat{\boldsymbol{z}})$ denote the bitrates of the quantized latent representation $\hat{\boldsymbol{y}}$ and the hyperprior $\hat{\boldsymbol{z}}$, respectively. The distortion $D(\boldsymbol{x}, \hat{\boldsymbol{x}})$ measures the reconstruction error between the original image $\boldsymbol{x}$ and the decoded image $\hat{\boldsymbol{x}}$, using either **MSE** or **MS-SSIM** as the quality metric. The hyperparameter λ controls the trade-off between compression rate and reconstruction fidelity.

For the **perceptual quality stage**, we adopt a multi-objective loss function to improve visual realism and mitigate common compression artifacts such as color distortion, over-smoothing, and structural blurring. The overall loss is defined as:

$$\mathcal{L} = R(\hat{\boldsymbol{y}}) + R(\hat{\boldsymbol{z}}) + \lambda \cdot (\mu_1 \cdot D_1 + \mu_2 \cdot D_2 + \mu_3 \cdot D_3), \tag{4}$$

where D_1 denotes the pixel-wise reconstruction error measured by **MSE**, D_2 represents the negative **MS-SSIM** (serving as a structural similarity loss), and D_3 is the **perceptual loss** based on deep features, such as **LPIPS** (Learned Perceptual Image Patch Similarity) [11]. The hyperparameters μ_1, μ_2, and μ_3 control the relative importance of each term, allowing the model to prioritize perceptual fidelity over pixel-wise accuracy, especially in low-bitrate regimes. This combined loss enables the network to generate visually pleasing reconstructions with preserved textures and natural appearance.

III. Implementation Details

Dataset. Our model is trained on a large-scale dataset comprising over 30,000 high-resolution images, including both natural photographs and screen content (e.g., text, graphics, UI elements). The natural image subset is selected from widely used benchmarks including **DIV2K** [12], **Flickr2K** [13], and **Flickr2W** [14], ensuring diverse scenes, textures, and lighting conditions. The screen content portion consists of self-captured screenshots and synthetic screen-like images, enhancing the model's generalization to mixed-content compression scenarios. All training images have resolutions exceeding 512×512. To mitigate potential artifacts from JPEG compression present in some sources, we apply a randomized downsampling strategy, ensuring the shorter side of each image falls within the range $[512, 584]$ pixels before cropping.

Training Setup. The framework is implemented in **PyTorch** and built upon the **CompressAI** library, which provides optimized operations for learned image compression. We use the **Adam optimizer** with momentum parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and an initial learning rate of 1×10^{-4}. All models are trained on a single **NVIDIA Tesla A100 GPU** with 40GB memory. The batch size is fixed at 32 throughout training. Input images are randomly cropped into 256×256 patches during the first 1.2 million steps to facilitate faster convergence. After this warm-up phase, we switch to 512×512 crops to better exploit long-range spatial dependencies and improve the model's ability to learn global context, which is particularly beneficial for high-resolution reconstruction and texture preservation.

Single-Rate Training. In the first stage, we train three individual models corresponding to low, medium, and high bitrates respectively, so as to obtain well-optimized checkpoints across the rate-distortion (R-D) curve, where distortion (D) only considers MSE. Each model corresponds to a specific λ value targeting a desired operating point, with $\lambda \in (5 \times 10^{-3}, 2 \times 10^{-2}, 1 \times 10^{-1})$. The learning rate is decayed at predefined milestones: reduced to 3×10^{-5} at 1.5M steps, 1×10^{-5} at 1.8M steps, 3×10^{-6} at 1.9M steps, and 1×10^{-6} at 1.95M steps. Training runs for a total of **2 million iterations**, ensuring convergence. These single-rate models serve as strong initialization points for subsequent variable-rate fine-tuning.

Variable-Rate Training with QP Embedding. Building upon the single-rate models, we introduce **quantization parameter (QP) embedding** to enable flexible variable-bitrate (VBR) control within a single unified network. The QP embedding is injected into both the main encoder and decoder, allowing dynamic adaptation to different bitrates at inference time without retraining. To densely sample the R-D curve, we select λ based on target bitrate regimes: for low-bitrate compression at approximately **0.075 BPP**, $\lambda \in (5 \times 10^{-3}, 1 \times 10^{-2})$; for medium-bitrate around **0.15 BPP**, $\lambda \in (1 \times 10^{-3}, 2 \times 10^{-2})$; and for high-bitrate near **0.3 BPP**, $\lambda \in (1 \times 10^{-2}, 1 \times 10^{-1})$. Three models spanning distinct rate ranges are trained across these λ interval, with the QP-aware network undergoing fine-tuning to enable continuous bitrate adaptation. Specifically, interpolation is performed over the λ interval based on the number of QPs, such that each QP is mapped to a unique λ value. In each iteration, a randomly selected QP-λ pair is employed to optimize the network. During inference, the target QP is provided as input, enabling seamless switching across bitrates.

Perceptual Quality Training. For models targeting high perceptual quality—particularly at low bitrates—we adopt the multi-objective loss defined in Eq. (4), which combines rate, distortion, and perceptual metrics. Specifically, D_1 is the **MSE** loss, D_2 is the negative **MS-SSIM** (i.e., $1-$MS-SSIM), and D_3 is the **LPIPS** loss computed using a pre-trained VGG network. The hyperparameters μ_1, μ_2, and μ_3 are set to 1.2, 0.08, and 0.02, respectively. This weighting emphasizes pixel-level accuracy while leveraging structural and semantic similarity to preserve fine textures, reduce blurriness, and suppress color artifacts. The perceptual models are trained using the same optimization and data pipeline, with a focus on low-bitrate regimes where visual fidelity is most challenging.

References

[1] C. Christopoulos, A. Skodras, and T. Ebrahimi, "The jpeg2000 still image coding system: an overview," *IEEE Trans. Consum. Electron.*, vol. 46, no. 4, pp. 1103–1127, Nov. 2000.

[2] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (hevc) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[3] B. Bross, Y.-K. Wang, Y. Ye, S. Liu, J. Chen, G. J. Sullivan, and J.-R. Ohm, “Overview of the versatile video coding (vvc) standard and its applications,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 10, pp. 3736–3764, Oct. 2021.

[4] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, “Variational image compression with a scale hyperprior,” in *Int. Conf. on Learn. Represent.(ICLR)*, Vancouver, Canada, Apr. 2018.

[5] S. Van Den Oord, A. Kalchbrenner, S. Colmenarejo, Z. Wang, Y. Chen, D. Belov, and Freitas, “Parallel multiscale autoregressive density estimation,” *International Conference on Machine Learning (ICML)*, 2017.

[6] D. Minnen, J. Ballé, and G. Toderici, “Joint autoregressive and hierarchical priors for learned image compression,” in *Adv. Neural Inf. Process. Syst. (NIPS)*, Montreal, Canada, Dec. 2018, pp. 10 794–10 803.

[7] H. Chang, H. Zhang, L. Jiang, C. Liu, and W. T. Freeman, “Maskgit: Masked generative image transformer,” in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Jun. 2022.

[8] D. Minnen and S. Singh, “Channel-wise autoregressive entropy models for learned image compression,” in *IEEE Int. Conf. Image Process. (ICIP)*, Abu Dhabi, United Arab Emirates, Oct. 2020, pp. 3339–3343.

[9] D. He, Y. Zheng, B. Sun, Y. Wang, and H. Qin, “Checkerboard context model for efficient learned image compression,” in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Jun. 2021, pp. 14 766–14 775.

[10] J. Li, B. Li, and Y. Lu, “Hybrid spatial-temporal entropy modelling for neural video compression,” in *ACM Int. Conf. Multimedia(MM)*, Lisboa, Portugal, 10 2022, p. 1503–1511.

[11] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, “The unreasonable effectiveness of deep features as a perceptual metric,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition(CVPR)*. IEEE, Jun. 2018, pp. 586–595.

[12] E. Agustsson and R. Timofte, “Ntire 2017 challenge on single image super-resolution: Dataset and study,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, Jul. 2017, pp. 1122–1131.

[13] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, “Enhanced deep residual networks for single image super-resolution,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, Jul. 2017, pp. 1132–1140.

[14] J. Liu, G. Lu, Z. Hu, and D. Xu, “A unified end-to-end framework for efficient deep image compression,” *arXiv preprint arXiv:2002.03370*, 2020.