Real-time 6K Image Rescaling with Rate-distortion Optimization

Chenyang Qi*  Xin Yang*  Ka Leong Cheng  Ying-Cong Chen  Qifeng Chen
Hong Kong University of Science and Technology

Abstract

Contemporary image rescaling aims at embedding a high-resolution (HR) image into a low-resolution (LR) thumbnail image that contains embedded information for HR image reconstruction. Unlike traditional image super-resolution, this enables high-fidelity HR image restoration faithful to the original one, given the embedded information in the LR thumbnail. However, state-of-the-art image rescaling methods do not optimize the LR image file size for efficient sharing and fall short of real-time performance for ultra-high-resolution (e.g., 6K) image reconstruction. To address these two challenges, we propose a novel framework (HyperThumbnail) for real-time 6K rate-distortion-aware image rescaling. Our framework first embeds an HR image into a JPEG LR thumbnail by an encoder with our proposed quantization prediction module, which minimizes the file size of the embedding LR JPEG thumbnail while maximizing HR reconstruction quality. Then, an efficient frequency-aware decoder reconstructs a high-fidelity HR image from the LR one in real time. Extensive experiments demonstrate that our framework outperforms previous image rescaling baselines in rate-distortion performance and can perform 6K image reconstruction in real time.

1. Introduction

With an increasing number of high-resolution (HR) images being produced and shared by users on the internet, a new challenge has arisen: how can we store and transfer HR images efficiently? Storing HR images on the cloud, such as iCloud, is becoming a widely adopted solution that saves storage on a user’s mobile device (e.g., smartphones) as only their low-resolution (LR) counterparts are stored on the mobile device for an instant preview. However, when a user wants to obtain the full-resolution image, the entire HR image must be downloaded on the fly from the cloud, which can result in a poor user experience when the internet connection is unstable or not available.

Real-time image rescaling can serve as a competitive solution to improving the user experience of cloud photo stor-

*Equal contribution.
between the thumbnail file size and the number of pixels. In this paper, we define the important and practical metric rarely studied in prior rescaling methods [36, 55].

To remedy these issues, our image rescaling framework is designed to jointly optimize image quality and bpp without explicit RD constraints. While some works [26, 47, 53, 54, 58] leverage the rate constraint by embedding extra information in JPEG, they simply utilize a fixed differentiable JPEG module, which we argue is sub-optimal for image rescaling. Because such a process deteriorates the information in the embedding images without considering their local distribution. Moreover, the quantization process of JPEG introduces noise in the frequency domain and introduces well-known JPEG artifacts, which brings great challenges to information restoration.

In this work, we propose the **HyperThumbnail**, a rate-distortion-aware framework for 6K real-time image rescaling, as shown in Table 1(c). In this framework, we embed an HR image into a low-bitrate JPEG thumbnail by an encoder and a quantization table predictor, as JPEG is a dominant image compression format today [3]. Then the JPEG thumbnail can be up-scaled to its high-fidelity HR counterpart with our efficient decoder in real time. We leverage an asymmetric encoder-decoder architecture, where most computation is put in the encoder to keep the decoder lightweight. This makes it possible for our decoder to upscale a thumbnail to 6K in real time, significantly faster than previous flow-based image rescaling methods [36, 55].

Meanwhile, the Rate-Distortion (RD) performance is an important and practical metric rarely studied in prior rescaling works. In this paper, we define the **rate** as the ratio between the thumbnail file size and the number of pixels in the HR image, also known as the bits-per-pixel (bpp). The **distortion** consists of two parts: the perceptual quality of the thumbnail (LR distortion) and the fidelity of the restored HR image (HR distortion). The rate-distortion performance evaluates an image rescaling framework in both storage cost and visual quality. Without explicit RD constraints, recent works in image rescaling [36, 55] do not consider RD performance in their models. While some works [26, 47, 53, 54, 58] leverage the rate constraint by embedding extra information in JPEG, they simply utilize a fixed differentiable JPEG module, which we argue is sub-optimal for image rescaling. Because such a process deteriorates the information in the embedding images without considering their local distribution. Moreover, the quantization process of JPEG introduces noise in the frequency domain and introduces well-known JPEG artifacts, which brings great challenges to information restoration.

Our contributions are summarized as follows:

- **We propose a 6K real-time rescaling framework with an asymmetric encoder-decoder architecture, named HyperThumbnail, which embeds a high-resolution image into a JPEG thumbnail that can be viewed in popular browsers. The decoder utilizes both spatial and frequency information to reconstruct high-fidelity images in real time for 6K image upsampling.**

- **We introduce a new quantization prediction module (QPM) that improves the RD performance in the en-

---

### Table 1. The comparison of different methods related to image rescaling.

<table>
<thead>
<tr>
<th>Method</th>
<th>(a) Downsampled JPEG + super-resolution [37]</th>
<th>(b) Flow-based rescaling [36, 55]</th>
<th>(c) Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architecture</td>
<td><img src="image.png" alt="Architecture" /></td>
<td><img src="image.png" alt="Architecture" /></td>
<td><img src="image.png" alt="Architecture" /></td>
</tr>
<tr>
<td>Reconstruction fidelity</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Rate-distortion optimization</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Real-time 6K reconstruction</td>
<td>–</td>
<td>✗</td>
<td>✓</td>
</tr>
</tbody>
</table>

(a) Super-resolution from downsampled JPEG does not optimize rate-distortion performance and can hardly maintain high fidelity due to information lost in downsampling. (b) SOTA flow-based image rescaling methods also ignore the file size constraints and are not real-time for 6K reconstruction due to the limited speed of invertible networks. (c) Our framework optimizes rate-distortion performance while maintaining high-fidelity and real-time 6K image rescaling.
coding stage of our framework. Furthermore, we adopt rate-distortion-aware loss functions along with QPM to optimize the RD performance.

- Experiments show that our framework outperforms state-of-the-art image rescaling methods with higher LR and HR image quality and faster reconstruction speed at similar file size.

2. Related Work

2.1. Image Super-resolution

Image super-resolution (SR) targets at restoring HR images from LR images. Pioneering works on SR such as SRCNN [17], EDSR [37], and other successors [14, 33, 34, 61, 62] exploit deep neural networks to solve the challenges in image SR. Recently, Liang et al. [35] propose a Swin Transformer-based framework with state-of-the-art image SR performance. In addition, Wang et al. [30, 51] extend SRGAN [31] and demonstrate the potential of producing perceptually pleasing HR images with a GAN-based generator. Despite great recent progress, most of these methods assume a simple deterministic downsampling process (e.g., Bicubic downsampling [40]), which limits the reconstruction quality of high-frequency details in image SR.

2.2. Image Rescaling

Different from SR, image rescaling aims to downsample the given HR image into a visually satisfying LR image with embedded information, and then reconstruct the HR image with high fidelity. A straightforward solution is downsampling using detail-preserving methods [29] and upsampling with heavy SR networks [14, 35, 37]. Recently, efforts have been made to apply invertible neural network (INN) to image rescaling. INN [9, 12, 15, 16, 19, 28, 30, 43] provides direct access to the inverse mapping of the forward function, making it a popular framework for image rescaling [13, 36, 55]. Xiao et al. [55] make the first attempt to model image downscaling and upsampling using invertible transformation. Liang et al. [36] further formulate the high-frequency components in INNs as a conditional distribution on LR image. However, the equivalence in the computational cost of the encoding and decoding process of INN makes it almost impossible to optimize the backward inference time independently, which limits its practicality in latency-aware scenarios. Besides, the file size of the embedding LR image in existing rescaling frameworks is yet to be studied. In this paper, we leverage the RD metric to evaluate the rescaling methods from a real-world perspective and compare our framework with existing image rescaling models in terms of their bpp and restored HR fidelity.

2.3. Image Compression

Instead of only shrinking the image size spatially, image compression approaches optimize the RD performance by first producing a compact bitstream, then decoding HR from the bitstream. Recently, learning-based methods [5–7, 21, 27, 32, 39, 44, 48, 56] have improved RD performance by a large margin with neural encoders and decoders. However, these methods have not been widely adopted on the internet due to issues related to software compatibility or runtime efficiency. If users of neural compression need a thumbnail for easier manipulation and preview, they need to save another redundant file besides the bitstream, which is inconvenient and takes extra storage. Thus, traditional compression algorithms (e.g., JPEG [49]) are still popular on most social media platforms [3].

In this work, we embed HR images into LR JPEG thumbnails and restore HR images with a 6K real-time decoder. Training neural networks with the non-differentiable JPEG algorithm is challenging. Previous works approximate JPEG either with the iterative optimization [26] or the differentiable degradation simulator [47, 63] during training, and revert to the conventional JPEG algorithm at test time. However, these methods do not optimize the RD performance in their models. In a concurrent work, Xiao et al. [54] extend IRN with a fixed JPEG compression module and an extra JPEG artifact removal module. However, they have not considered the bpp and the reconstruction quality as a joint optimization problem. In this work, we propose a novel differentiable JPEG process with the QPM guided by a bitrate loss for RD performance optimization. Moreover, our real-time decoder reconstructs the HR image from both the spatial and frequency domains of a thumbnail and further improves the reconstruction quality.

3. Method

3.1. JPEG Preliminary

As our solution involves optimizing the rate-distortion (RD) performance of the JPEG thumbnail, we briefly summarize the JPEG algorithm [49] in this section. The JPEG algorithm compresses an image in three steps. First, given an RGB image \( y \in \mathbb{R}^{3 \times h \times w} \), the algorithm converts \( y \) into the luma-chroma color space (YCbCr). Second, the converted image is divided into 8 \( \times 8 \) pixel blocks as \( y \in \mathbb{R}^{3 \times N \times 8 \times 8} \), where \( N = \frac{h \times w}{8} \). Then, \( y \) is transformed to its corresponding Discrete-Cosine-Transform (DCT) coefficients \( C \in \mathbb{R}^{3 \times N \times 8 \times 8} \):

\[
C = (C_Y, C_{Cb}, C_{Cr}) = \text{DCT}(y_Y, y_{Cb}, y_{Cr}).
\]

Third, the luma coefficient \( C_Y \) and chroma coefficient \( C_C = (C_{Cb}, C_{Cr}) \) are quantized separately by two quan-
Given an HR input image $x$, we first encode $x$ to its LR representation $y$ with the encoder $E$, where the scaling factor is $s$. Second, we transform $y$ to DCT coefficients $C$ and predict the quantization tables $Q_L$, $Q_C$ with our quantization prediction module (QPM). Third, we adopt an entropy model [6] to estimate the bitrate of the quantized coefficients $\hat{C}$ at training stage. After the rounding and truncation, which we denoted as $\lfloor \cdot \rceil$, the $[Q_L]$, $[Q_C]$ and $[\hat{C}]$ can be written and read with off-the-shelf JPEG API at the testing stage. To restore the HR, we extract features from $\hat{C}$ with a frequency feature extractor $f$ and produce the high-fidelity image $\hat{x}$ with the decoder $D$.

The overview of our approach. Given an HR input image $x$, we first encode $x$ to its LR representation $y$ with the encoder $E$, where the scaling factor is $s$. Second, we transform $y$ to DCT coefficients $C$ and predict the quantization tables $Q_L$, $Q_C$ with our quantization prediction module (QPM). Third, we adopt an entropy model [6] to estimate the bitrate of the quantized coefficients $\hat{C}$ at training stage. After the rounding and truncation, which we denoted as $\lfloor \cdot \rceil$, the $[Q_L]$, $[Q_C]$ and $[\hat{C}]$ can be written and read with off-the-shelf JPEG API, at the testing stage. To restore the HR, we extract features from $\hat{C}$ with a frequency feature extractor $f$ and produce the high-fidelity image $\hat{x}$ with the decoder $D$.

The overview of our framework. Given an HR input image $x \in \mathbb{R}^{3 \times H \times W}$, we first generate its LR representation $y \in \mathbb{R}^{3 \times \frac{H}{s} \times \frac{W}{s}}$ through our encoder $E$, where the $s$ is the rescaling factor and our encoder $E$ is an U-Net [45] with dense blocks [23]. To further decrease the file size, we transform $y$ to DCT coefficients $C \in \mathbb{R}^{3 \times \frac{H}{s} \times \frac{W}{s} \times 8 \times 8}$. Then, we quantize $C$ with image-specific quantization tables $Q \in \mathbb{R}^{2 \times 8 \times 8}$ (Sec. 3.3). During the test time, we encode and decode $(Q, C)$ with off-the-shelf JPEG API, and then retrieve $(\hat{C}, \hat{y})$ with Eqn. (3). In the upsampling stage, we reconstruct the high-fidelity HR $\hat{x}$ with our efficient frequency-aware decoder in real time (Sec. 3.4).

### 3.3. Quantization Prediction Module

As JPEG [49] is a widely adopted compression algorithm, some previous methods [26, 47, 58, 63] in image embedding also introduce JPEG (Sec. 3.1) to compress their encoded image. However, JPEG degrades the embedded information in the encoded image and leads to a performance drop in the decoding stage (Sec. 5). JPEG uses fixed quantization tables with constant values [49], which we believe is suboptimal since different images have different frequency distribution. Hence, besides encoding HR images to LR thumbnails, we also predict the quantization tables $Q$ for each thumbnail with the quantization prediction module.
(QPM), which boosts the RD performance of our framework significantly. The QPM is implemented as two separate 8-layer multilayer perceptron (MLPs): the luma predictor MLP_L and the chroma predictor MLP_C. For block \( C_k = (C_{Y,k}, C_{C,k}) \), we vectorize \( C_k \) into a 1D vector and produce its quantization table with QPM. Thus we have

\[
Q_L = \frac{\sum_k \text{MLP}_L(C_{Y,k})}{|C_{Y,k}|}, Q_C = \frac{\sum_k \text{MLP}_C(C_{C,k})}{|C_{C,k}|},
\]

where \(| \cdot |\) denotes the block counts. To facilitate the conventional JPEG codec, we take the average \( Q \) as the quantization table for the whole image. Following the training stage of learned compression [7], we adopt the additive uniform noise \( \epsilon \) where each element follows \( U(-\frac{1}{2}, \frac{1}{2}) \) to approximate the non-differentiable quantization noise introduced in Eqn. (2):

\[
\tilde{C}_Y = \frac{C_Y}{Q_L + \epsilon}, \quad \tilde{C}_C = \frac{C_C}{Q_C + \epsilon} + \epsilon.
\]

In the testing stage, we switch back to the standard quantization function in Eqn. (2) to fit the JPEG API. Experiments show that our QPM improves the RD performance of our framework (Sec. 5). Our experiments in the supplement also show that our QPM improves the standard JPEG algorithm.

### 3.4. Frequency-aware Decoder

As mentioned in Sec. 3.1, the JPEG quantization is applied on the DCT domain coefficients \( C \) and leads to the well-known quantization noise. Previous works [47, 53, 58] usually decode the embedded information from the RGB domain of the JPEG image. However, we note that quantization noise can be approximated by simple independent uniform noise [6] in the DCT domain, but it is nonlinear and more complex to model in the RGB domain after IDCT.

Inspired by recent works in JPEG artifact removal [18, 20, 52, 60] and image generation [42], we further propose a novel plug-in named frequency feature extractor \( f \) for additional frequency domain perception. We first reshape the dequantized DCT coefficients into a sparse representation \( \tilde{C} \in \mathbb{R}^{192 \times \frac{W}{s} \times \frac{H}{s}} \) [42]. Then, we extract features \( f(\tilde{C}) \in \mathbb{R}^{24 \times \frac{W}{s} \times \frac{H}{s}} \) and concatenate \( f(\tilde{C}) \) with the RGB image \( \hat{y} \). The concatenated features are fed into the decoder to reconstruct the HR image \( \hat{x} \):

\[
\hat{x} = D(\hat{y} \oplus f(\tilde{C})),
\]

where \( \oplus \) is the concatenation operator along the channel dimension. Since \( \tilde{C} \) is the \( \frac{1}{s} \) of \( \hat{y} \) in the spatial resolution, \( f \) takes negligible computational overhead. In this way, our decoder is lightweight enough to reconstruct 6K HR images in real time (Fig. 5). \( f \) improves our RD performance and is more efficient than increasing the decoder capacity. Please refer to our supplement for more details of our architecture.

### 3.5. Training Objectives

**Bitrate loss** To optimize the RD performance of our framework, it is critical to estimate the code length (bitrate) of the quantized coefficients \( \tilde{C} \) at the training stage. Inspired by Ballé et al. [6], the rate \( R \) of \( \tilde{C} \) is estimated with differentiable fully-factorized entropy models:

\[
R = E_{x \sim p_x}[-\log_2 p_L(\tilde{C}_Y) - \log_2 p_C(\tilde{C}_C)] = \log_2 p_L(\tilde{C}_Y) - \log_2 p_C(\tilde{C}_C),
\]

where \( p_L \) and \( p_C \) are two fully-factorized entropy models for luma and chroma coefficient maps, respectively. In our work, we reshape \( \tilde{C} \) to \( (3, 64, \frac{H}{8s}, \frac{W}{8s}) \), and assume the value of each pixel in \( \tilde{C} \) to be independent following Ballé et al. [6]. For accurate rate estimation, we train \( p_L \) and \( p_C \) to model the 64-channel probability density functions of \( \tilde{C}_Y \) and \( \tilde{C}_C \) by minimizing an auxiliary loss \( L_{aux} \) [7]. The bpp of the restored image \( \hat{x} \) is calculated by

\[
L_{bpp} = \frac{R}{H \times W}.
\]

More details of auxiliary loss is provided in the supplement.

**Reconstruction and guidance loss** Thanks to the fully differentiable pipeline (Fig. 2), it is possible for us to train our encoder, QPM and decoder with similar loss terms used in previous works [36, 55]. Following IRN [55], we employ \( L_1 \) on the reconstructed \( \hat{x} \) and a \( L_2 \) guidance loss on the JPEG thumbnail \( \hat{y} \):

\[
L_{recon} = \frac{||\hat{x} - x||_1}{H \times W}, \quad L_{guide} = \frac{||\hat{y} - y_{ref}||_2}{(H/s) \times (W/s)}.
\]

where \( y_{ref} \) is a guidance image downscaled from \( x \) with bicubic interpolation [40]. Altogether, we train our framework by minimizing the total loss \( L_{rescale} \):

\[
L_{rescale} = L_{recon} + \lambda_1 L_{guide} + \lambda_2 L_{bpp}.
\]

Empirically, we set \( \lambda_1 = 0.6 \). The target bpp of our framework could be adjusted by a loss scaling factor \( \lambda_2 \), which we set to 0.01 for most experiments. The total loss \( L_{rescale} \) is adopted to optimize the parameters of the encoder, the decoder, and QPM.

### 4. Experiments

#### 4.1. Experimental Setup

**Evaluation metrics** We evaluate our upscaling efficiency using running time and multiply-accumulation operations (MACs). We evaluate the quality of our reconstructed and embedding images using the PSNR on RGB channels.

Rescaling methods are developed to embed an HR image into an LR one to reduce file size. However, there is no existing comprehensive evaluation of bitrate for these methods. Following previous compression approaches [6, 7, 56],
Table 2. Quantitative evaluation of upscaling efficiency and reconstruction fidelity. We keep bpp around 0.3 on Kodak [1] for different methods, and the distortion is measured by the PSNR on the reconstructed HR images. Our approach outperforms other methods with better HR reconstruction and a significantly lower runtime. We measure the running time and GMacs of all models by upscaling a 960 \times 540 LR image to a 3840 \times 2160 HR image. The measurements are made on an Nvidia RTX 3090 GPU with PyTorch-1.11.0 in half-precision mode for a fair comparison.

![Figure 3. Reconstructed HR images and LR thumbnails by different methods on the DIV2K [4] validation dataset. We crop the restored HR images to ease the comparison and visualize the LR counterparts at the bottom-right. The bpp is calculated on the whole image and the PSNR is evaluated on the cropped area of the reconstructed HR images.](image)

we evaluate the RD performance with the HR reconstruction PSNR against bits-per-pixel. Instead of estimating the bitrate from the entropy encoder (Eqn. (7)) during training, we use the real file size of JPEG thumbnails for evaluating the bitrate:

\[
bpp = \mathbb{E}_{x \sim p_x} \frac{\text{file size}}{H \times W}.
\]

Since there exists a trade-off between image bitrate and fidelity, we visualize the rate-distortion curve of different models during our ablation study.

Datasets We train our model with the widely-used DIV2K [4] image dataset, which contains 800 2K-resolution images in the training set and 100 more in the validation set. From the perspective of rescaling, we test our model on 4 conventional datasets: the Set5 [10], Set14 [59], BSD100 [38], and Urban100 [24]. To reveal our real-world performance, we collect first 20 images above 6K resolution (ranked by their bpp) from MIT-Adobe FiveK [11] as another test set FiveK-6k. In addition, we evaluate the RD performance on Kodak dataset [1], which is widely adopted in image compression researches [8]. The details of the training strategy are provided in the supplement.

4.2. Compare with Baselines

We consider three categories of reconstruction methods as our baselines: (1) downsampling with Bicubic interpolation, compressing with standard JPEG codec, and upsampling with state-of-the-art SR models [35, 37]; (2) autoencoder using standard JPEG transmission format [26]; (3) rescaling using symmetric invertible neural network [36, 55] and compressing the LR image using standard JPEG codec. For fair evaluation, we improve the RD performance of baseline using standard JPEG compression. We retrain SR models [35, 37] on LR JPEG compressed images to restore
HR images. Besides, we retrain INN baselines [36, 55] and encoder-decoder baselines [26] with a differentiable JPEG module [46]. To compare our RD performance against baselines, we constrain their bpp on Kodak dataset to be around 0.3 by adjusting the quality factor of JPEG compression in all baselines. In our supplement, we provide an additional comparison with image compression (e.g., the original JPEG) and rescaling baselines using their original transmission format (e.g., lossless PNG), where our advantage is even larger.

**Upscaling efficiency and HR fidelity** Table 2 presents the upscaling efficiency of all methods and reconstruction fidelity at around 0.3 bpp. We measure the running time and GMacs of upscaling a 960 × 540 resolution LR image to a 3840 × 2160 HR image on an Nvidia RTX 3090 GPU. We use PyTorch implementation with 16 bits floating-point precision for a fair comparison. As shown in Table 2, “Ours” model only costs 3.1% of the GMacs and 3.7% of upscaling time compared to HCFlow [36]. We still improve the reconstructed RGB PSNR by 0.61 dB on BSD100 test set with a significantly lower computation.

In addition, we train the “Ours-full” model with a larger decoder. “Ours-full” model achieves higher PSNR that outperforms HCFlow [36] and IRN [55] for more than 1.16 dB on Urban100 test set with only 24.2% of upscaling time. Fig. 3 provides the visual comparison of restored images. From left to right, we show perceptual differences from ground truth, 4x rescaling results of the baselines [36, 55] with JPEG, and our framework. For the first row, our models restore more textures of the glass and the pot lid. Similarly, among all reconstructions in the second row, only our model can recover the details on hair and earmuffs. Besides, the JPEG compression breaks the invertibility of the INN-based rescaling methods [36, 55], they failed to restore sharp and accurate high-frequency textures in the HR images.

**LR qualitative evaluation** The visual quality of the LR thumbnails is also important because users preview them directly. Table 3 presents the average LR PSNR of corresponding models, we have the best LR PSNR against INN-based methods. Fig. 4, we qualitatively compare the visual quality of the thumbnails encoded by rescaling methods. Our method generates LR thumbnails with significantly fewer artifacts without compromising much HR reconstruction quality, please see more analysis in Sec. 5.

### 4.3. Real-time Inference on 6K Images

Fig. 5 shows our evaluation of the downsampling and upsampling runtime of our framework at multiple resolutions. Noted that we conduct the profiling in half-precision (FP16) mode. Besides, we also converted the trained decoder model from PyTorch to FP16 TensorRT model with torch2trt [2] API to further reduce the upsampling time without performance drop in HR reconstruction. As shown in Fig. 5(b), our efficient decoder can upscale an LR thumbnail by 4x in real time at 4K (3840 × 2160) **70.8** FPS, 5K (5120 × 2880) **38.8** FPS, or 6K (5760 × 3240) **31.2** FPS on an RTX 3090 GPU.

### 4.4. Extension for Optimization-based Rescaling

Since the ground truth HR images are available for downsampling during the test stage, we may further optimize our encoder E, QPM, and Entropy model, while fixing the pretrained decoder D and feature extractor f on the user’s device. As shown in the Table 4, optimization-based rescaling improves the restoration PSNR 0.22dB with a lower bitrate on Set14 dataset.
Quantization prediction module To examine the effectiveness of our QPM, in Fig. 7, we quantitatively evaluate the RD performance on both the restored HR and the LR JPEG thumbnails of different quantization approaches. For “fixed tables” and “optimized tables”, we initialize the quantization tables following the default JPEG. Particularly, for “optimized tables,” we also optimize the quantization tables at the training stage. We illustrate the curve for each model by adjusting the global quality factor \( q \) on the quantization tables \( Q \) following JPEG as \( Q' = Q \times q \). The target bitrate goes lower when \( q \) increases. In Fig. 6, we also visualize the quantization tables of different settings. We notice that compared to the corresponding value in the fixed or optimized table, the high-frequency quantization steps in the QPM predicted tables are much smaller and are image-specific, which may introduce less compression on the embedding pattern that is important for HR reconstruction. Consequently, compared to settings with image-invariant quantization, “Ours” model achieves the best quality on the reconstructed \( x \) and introduces significantly fewer artifacts on the LR thumbnail \( y \), which also lowers the file-size of the thumbnails. To further investigate the effectiveness of QPM, we also leverage QPM to improve the RD performance of standard JPEG (without the downscaling and upscaling). Please refer to our supplement for more details.

Guidance loss In Fig. 7, we also present the RD curve of the “Ours w/o guidance loss” model trained with \( \lambda_1 = 0 \). At around 0.4 bpp, removing guidance loss raises the HR PSNR by 0.09dB. However, the PSNR of LR thumbnail drops significantly from 34.1 dB to 13.06 dB. It is unacceptable for user-viewable thumbnails.

Encoder decoder architecture As we study some variants of “Ours” model, we find that \( f \) improves the quality of both encoded LR thumbnail (0.29 dB) and restored HR image (0.10 dB). Also, adopting \( f \) is more effective than simply increasing the decoder capacity. Moreover, our study of the encoder capacity reveals that our framework does benefit from a larger encoder. More details are in our supplement.

6. Conclusion

In this paper, we propose a new HyperThumbnail framework that can perform real-time 6K image reconstruction from an LR JPEG thumbnail. We utilize an asymmetric encoder-decoder architecture where the encoder takes most of the computation while the decoder is relatively lightweight for real-time performance. A new quantization prediction module is proposed to optimize the RD performance for image rescaling, which is not studied in prior work. Our framework benefits image sharing and transfer in real-world latency-sensitive applications, such as cloud photo storage and retrieval.
References


[22] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B. Girshick. Masked autoencoders are scalable vision learners. In Proceedings of CVPR, 2022. 4


