Learned Image Compression with Dual-Branch Encoder and Conditional Information Coding

Haisheng Fu[*†], Feng Liang[*], Jie Liang[†], Zhenman Fang[†],

Guohe Zhang[*], and Jingning Han[⋆]

[*]Xi'an Jiaotong University [†]Simon Fraser University [⋆]Google

Abstract

Recent advancements in deep learning-based image compression are notable. However, prevalent schemes that employ a serial context-adaptive entropy model to enhance rate-distortion (R-D) performance are markedly slow. Furthermore, the complexities of the encoding and decoding networks are substantially high, rendering them unsuitable for some practical applications. In this paper, we propose two techniques to balance the trade-off between complexity and performance. First, we introduce two branching coding networks to independently learn a low-resolution latent representation and a high-resolution latent representation of the input image, discriminatively representing the global and local information therein. Second, we utilize the high-resolution latent representation as conditional information for the low-resolution latent representation, furnishing it with global information, thus aiding in the reduction of redundancy between low-resolution information. We do not utilize any serial entropy models. Instead, we employ a parallel channel-wise auto-regressive entropy model for encoding and decoding low-resolution and high-resolution latent representations. Experiments demonstrate that our method is approximately twice as fast in both encoding and decoding compared to the parallelizable checkerboard context model, and it also achieves a 1.2% improvement in R-D performance compared to state-of-the-art learned image compression schemes. Our method also outperforms classical image codecs including H.266/VVC-intra (4:4:4) and some recent learned methods in rate-distortion performance, as validated by both PSNR and MS-SSIM metrics on the Kodak dataset.

Introduction

Image compression is a fundamental and crucial topic in the field of signal processing. Over the past few decades, several classical standards have emerged, including JPEG [1], JPEG2000 [2], BPG [3], and VVC, which generally follow the same transform coding paradigm: linear transformation, quantization, and entropy encoding.

Recent learned image compression methods [4, 5, 6] have outperformed the current best classical image and video encoding standard VVC in terms of both peak signal-to-noise ratio (PSNR) and multi-scale structural similarity (MS-SSIM). This indicates that learned image compression methods hold tremendous potential for the next generation of image compression technologies.

Most learning-based image compression methods are Convolutional Neural Networks-based (CNN-based) approaches [7, 5, 8] that use the variational autoencoder (VAE) [9]. However, with the recent development of vision Transformers [10], several transformer-based learning methods [11, 12, 13] have been introduced. For instance, in CNN-based approaches, a residual block-based image compression model is proposed to achieve performance comparable to VVC in terms of PSNR. Meanwhile, in the realm of transformer-based methods, a swin-transformer-based image compression model has been proposed to enhance rate-distortion performance. Both CNN-based and transformer-based approaches offer distinct advantages: CNNs excel in local modeling with lower complexities, whereas transformers are adept at capturing non-local information. Nevertheless, the complexity of swin-transformer-based methods outperforms that of the CNN schemes.

The design of entropy models is also a critical aspect of learned image compression. A common approach introduces additional latent variables as hyper-priors, thereby transforming the compact encoded symbol probability model into a joint model [9]. Based on this, several methods [14, 7, 4] have been developed. For instance, masked convolution [14] is proposed to capture context information. More accurate probabilistic models, such as GMM [7] and GLLMM [4], have been introduced to enhance compression performance. Furthermore, a parallel channel autoregressive entropy model has been proposed [15], wherein the latent representation is divided into 10 slices. The encoded slices can aid the encoding of subsequent slices by providing side information in a pipelined manner.

Recently, some attention modules [16, 7] have been proposed to enhance image compression. Attention modules can be incorporated into the image compression framework to assist the model in focusing more on detailed information. However, many schemes are time-consuming or can only capture local information [16]. To reduce the complexity of the attention module, a simplified attention model is placed in the main encoder and decoder to enhance image compression. We also introduce an attention module [7] to improve the rate-distortion performance.

The contributions of this paper can be summarized as follows:

First, we employ two branches of encoders to learn latent representations at different resolutions of the input. This strategy allows us to better capture both global and local information from the input image. Notably, each branch operates independently without any information exchange.

Second, we utilize the high-resolution latent representation as conditional information to assist in the encoding and decoding of the low-resolution latent representation. This strategy helps to remove spatial redundancy in the low-resolution latent representation, enhancing the overall performance of the framework. We do not utilize any serial entropy models. Instead, we employ a parallel channel-wise auto-regressive entropy model [15, 13] for encoding and decoding low-resolution and high-resolution latent representations.

Third, extensive experiments demonstrate that our method achieves state-of-the-art performance when compared to recent learning-based methods and traditional image codecs on the Kodak dataset. Our method is approximately twice as fast in both encoding and decoding compared to the parallelizable checkerboard context model [6], and it also achieves a 1.2% improvement in R-D performance compared to state-of-the-art learned image compression schemes. Our method also outperforms the latest classical image codec in H.266/VVC-Intra (4:4:4) and other leading learned schemes such as [6] in both PSNR and MS-SSIM metrics. The decoding time and BD-Rate comparisons with VVC for various methods are presented in Fig. 1.

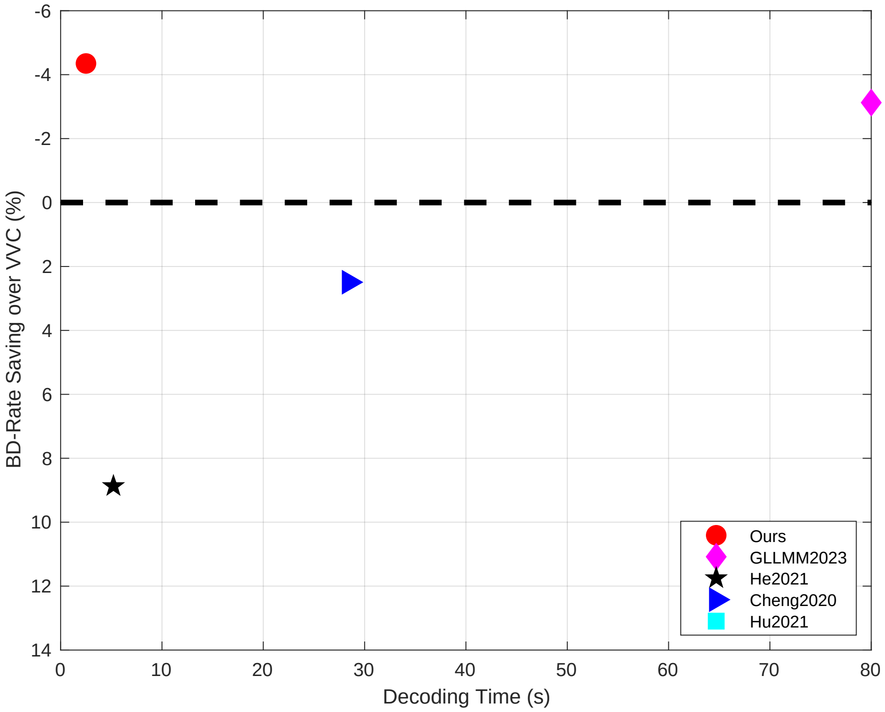

Figure 1: The decoding time and BD-Rate savings over H.266/VVC for different schemes are illustrated on Kodak dataset. A superior result is positioned in the upper-left corner. The notably extended decoding time for GLLMM [4] is explicitly indicated in brackets.

1 The Proposed Image Compression Framework

In this section, we describe the whole framework of the proposed method. Subsequently, we will detail its major components, including the dual-branch encoding network, conditional coding of low-resolution latent representations, and the associated training methodology.

The proposed framework is illustrated in Fig. 2. The input image, represented by x, has dimensions $W \times H \times 3$, where W and H are its width and height, respectively. The framework primarily consists of the core networks (g_{a1}, g_{a2} and g_s) and the hyper networks (h_a and h_s).

The two core encoder networks, labeled as g_{a1} and g_{a2}, are tasked with learning two compact latent representations y_1 and y_2 from the input image. The architectures of g_{a1} and g_{a2} both integrate two simplified attention modules, three residual group blocks (highlighted in cyan in Fig. 2), and four stages of pooling operations. The residual group blocks are composed of four basic residual blocks [17] connected in series.

We employ two branches to capture different resolutions of the original image. The first branch learns the high-resolution latent representation y_1, utilizing a 3×3 convolution as its downsampling module. In contrast, the second branch captures the low-resolution latent representation y_2 of the input image, leveraging a 1×1 convolution for downsampling.

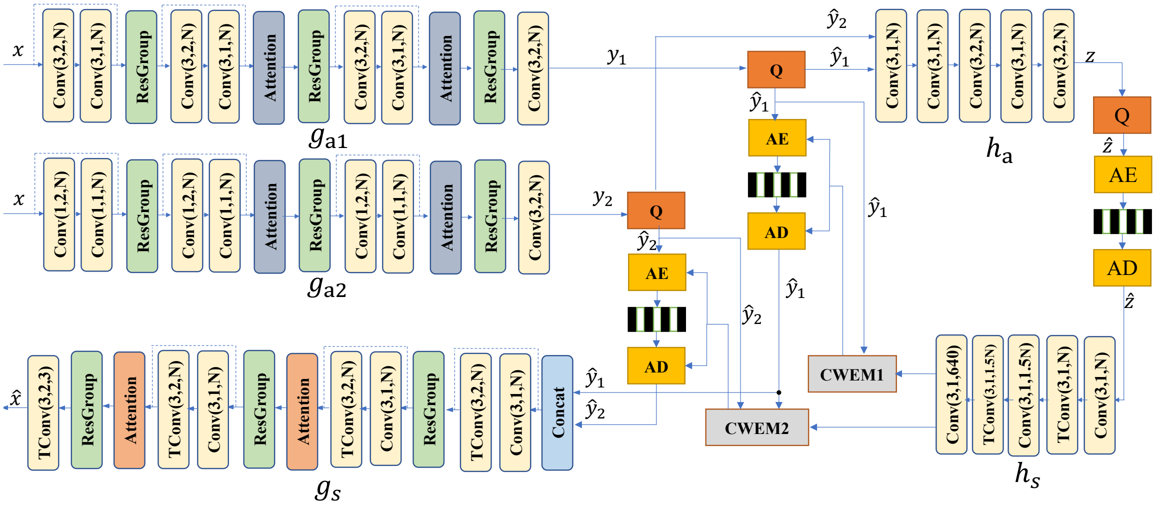

Figure 2: The overall architecture of our image compression framework. **CWEM** represents channel-wise Entropy Model. **ResGroup** represents the residual group blocks.

To enable parallel entropy decoding of the quantized latents y_1 and y_2, we do not use any serial context model. As in [15, 13], we use channel-wise entropy model to encode and decode y_1 and y_2 in parallel. However, it's noted that we first encode and decode y_1 in parallel, and then use y_1 as conditional side information to encode and decode y_2 in parallel. We will provide a detailed explanation of this encoding and decoding process in Sec. 1.2 and illustrate it in Fig. 3.

Subsequently, arithmetic coding compresses $\hat{y_1}$ and $\hat{y_2}$ into a bitstream. The decoded values, $\hat{y_1}$ and $\hat{y_2}$, are concatenated and forwarded to the primary decoder network g_s. This decoder mirrors the core encoder network g_a, with convolutions replaced by deconvolutions. While most convolution layers employ the leaky ReLU activation function, the final layer in both the hyperprior encoder and decoder operates without any activation function

1.1 Dual-Branch Main Encoder Networks

We use two separate encoding networks to learn different resolution latent representations of the input image, and these two branch encoding networks do not share any information. However, it is important to note that the sizes of the downsampling convolution kernels we use are different, ensuring that they can capture distinct information from the input image. Larger convolution kernels are capable of learning global information from the input image, while smaller convolutions can capture local information, making it easier to reduce spatial redundancy in the data.

1.2 Channel-Wise Auto-Regressive Entropy Model Based on Conditional Information Coding

We use two encoding networks to learn latent representations of the input image at different resolutions, denoted as y_1 and y_2. The first encoding sub-network utilizes larger convolution kernels to capture the global information from the input image, preserving information that is more global in nature in the latent representation. The second encoding sub-network uses smaller convolution kernels to focus on the local information of the input image.

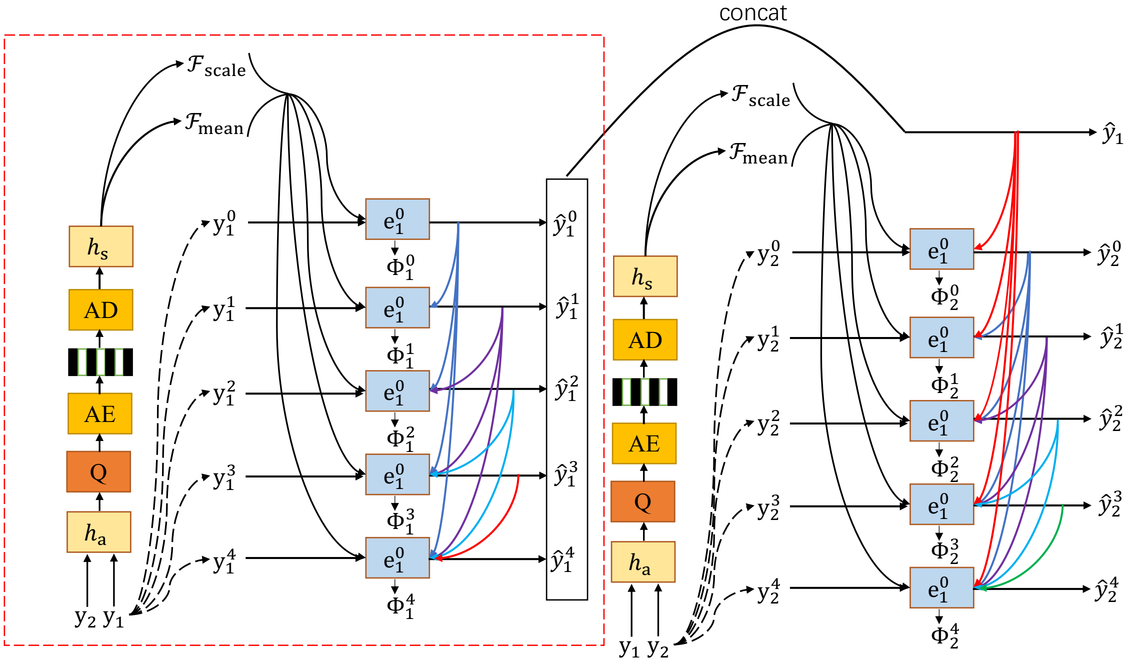

Figure 3: The channel-wise auto-regressive entropy model.The network of the ϕ have the similar networks where we just remove swin-transformer-based attention (SWAtten) modules from ϕ.

We use y_1 as auxiliary information to provide side information to y_2, thereby enhancing the efficiency of encoding and decoding for y_2. Since y_1 can provide y_2 with global information, it helps to eliminate redundancy in y_2.

As in [15, 13], we also use the channel-wise auto-regressive entropy model to encode and decode y_1 and y_2. The detailed processing is shown in Fig. 3. As in [13], We evenly divide the channels of y_1 and y_2 into five slices. The channel number of y_1 and y_2 are fixed at 320, so each slice has 64 channels. Our channels are encoded and decoded in sequence, where later channels can fully utilize the information from preceding channels as prior knowledge, thereby reducing spatial redundancy between channels. When encoding and decoding the information of y_2, we can incorporate the information from y_1 into the encoding process of y_2. Every encoded or decoded slice can make use of y_1 as conditional side information.

During encoding and training, we can obtain the $\hat{y}_1$ and $\hat{y}_2$ in parallel. Since the values of $\hat{y}$ are available during these phases, encoding and training of $\hat{y}_1$ and $\hat{y}_2$ can proceed concurrently.

However, during decoding, as we cannot access all values of latent representations $\hat{y}_1$ and $\hat{y}_2$ simultaneously, we must decode them sequentially. Given that a subsequent slice relies on information from the preceding slice, these slices are decoded in sequence. However, the individual elements within each slice can be decoded in parallel.

Finally, we can combine $\hat{y}_1$ and $\hat{y}_2$ to obtain the decoded $\hat{y}$.

1.3 Training

The training images are obtained from the CLIC dataset [1] and the LIU4K dataset [18]. These images are randomly resized to a resolution of 2000 × 2000. Through data augmentation methods, such as rotation and scaling, we generate a collection of 81,650 training images, each with a resolution of 384 × 384.

Our proposed models are optimized using two distortion metrics: mean squared error (MSE) and multi-scale structural similarity (MS-SSIM). For the MSE-optimized, we choose λ_1 values from the set $0.0016, 0.0032, 0.0075, 0.015, 0.03, 0.045, 0.06$. Each selected λ initiates the training of a distinct model tailored for a particular bit rate. The filter number N for latent representation is set at 128 for all bit rates. For MS-SSIM, λ sequentially takes on values 12, 40, 80, and 120. All λ values N remains 128. Each model undergoes 1.5×10^6 training iterations using the Adam optimizer with a batch size of 8. The starting learning rate is set at 1×10^{-4} for the first 750,000 iterations, then it gets halved after every subsequent 100,000 iterations.

2 Experimental Results

In this section, we compare some recent learned image compression methods and traditional image codecs with our proposed method in terms of Peak Signal-to-Noise Ratio (PSNR) and MS-SSIM metrics. The performance of different schemes are evaluated in two datasets with different resolutions. The Kodak PhotoCD dataset [2] is comprised of 24 images with a resolution of 768 × 512. Some recent learning-based schemes includes GLLMM [4], He2021 [6], Hu2021 [19], and Cheng2020 [20]. The traditional image codecs includes the latest image codec VVC-Intra (4:4:4) [3], BPG-Intra (4:4:4), JPEG2000, and JPEG.

For a fair comparison, we have implemented the method described in Cheng2020 [20], increasing its number of filters N from 192 to 256 at high rates. This modification leads to enhanced performance compared to the original results in [20]. The results from He2021 [6] are sourced from the code available at [21].

2.1 Rate-Distortion Performances

Fig. 4 depicts the average R-D curves of various methods evaluated on the Kodak dataset. Among the PSNR-optimized methods, GLLMM (MSE) [4] achieves the best performance in other methods, surpassing even VVC (4:4:4). Our method closely matches the coding performance of GLLMM at high bit rates and achieves better

[1] `http://www.compression.cc/`

[2] `http://r0k.us/graphics/kodak/`

[3] `https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM/tree/VTM-5.2`

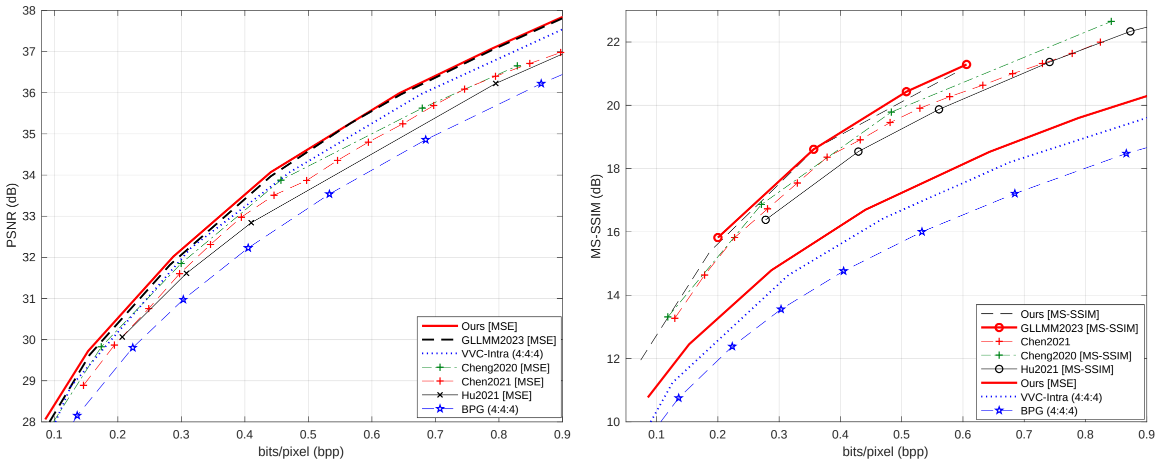

Figure 4: The average PSNR and MS-SSIM performance of the 24 images in the Kodak dataset.

Table 1: Comparisons of encoding and decoding time, BD-Rate saving over VVC, and model sizes of the low bit rates and high bit rates.

	Method	**Enc.**	**Dec.**	**BD-Rate**	**Model(L)**	**Model(H)**
Kodak	VVC	402.3s	0.61s	0.0	7.2 MB	7.2MB
	Hu2021 [22]	35.7s	77.3s	11.1 %	84.6 MB	290.9MB
	Cheng2020 [7]	26.4s	28.5s	2.6 %	50.8 MB	175.2MB
	He2021 [6]	24.4s	5.2s	8.9 %	46.6 MB	156.6 MB
	GLLMM [4]	467.9s	467.9s	-3.13%	77.1 MB	241.0MB
	Ours	**2.2 s**	**2.5s**	**-4.35%**	**68.6 MB**	**68.6MB**

performance at low bit rates. Our method has a 0.3-0.35 dB gain over VVC (4:4:4) at all bit rates. Regarding MS-SSIM, our method slightly outperforms GLLMM.

2.2 Complexity and Performance Trade-off

Table 1 illustrates a comparative results of average encoding/decoding times, BD-Rate savings relative to VVC [23], and model sizes at both low and high bit rates across various methods. Due to the non-deterministic issues encountered in GLLMM [4], VVC, Hu2021 [19], and Cheng2020 [7] when executed on GPU, we conducted evaluations exclusively on a common CPU platform, namely the 2.9GHz Intel Xeon Gold 6226R CPU, to ensure fairness in the comparisons.

Compared to the GLLMM [4], our method is much faster in encoding and decoding, about 200 times quicker. Our model works better and is smaller in size.

Compared to Cheng2020 [7], our method is much faster in both encoding and decoding, being approximately 11 times quicker. Additionally, our rate-distortion performance outperforms that of Cheng2020 by 6.95%. Compared to [6], not only is our encoding and decoding speed superior, but our R-D performance also shows a significant improvement, outperforming it by roughly 13.25%.

2.3 Performance Improvement of Different Modules

Table 2: The performance of different modules

Module	**Bit rate**	**PSNR**	**MS-SSIM**
Ours	0.1531	29.72 dB	12.67 dB
w/o CI	0.1582	29.63 dB	12.59 dB
w/o TB	0.1632	29.51 dB	12.48 dB
Ours	0.9013	37.85 dB	20.53 dB
w/o CI	0.9123	35.78 dB	20.48 dB
w/o TB	0.9146	35.62 dB	20.36 dB

Table 2 compares the results when we do use conditional information (CI) and two branches (TB) main encoder respectively, and the other modules remain the same. It can be seen that the performance drops by about 0.1 dB without conditional information (CI). The performance drops by about 0.2 dB at both low and high bit rates without two branches (TB).

2.4 Performance Comparison of Different Group Partitions

Table 3: The performance of different groups

Groups	**Bit rate**	**PSNR**	**MS-SSIM**	**model Size**
5	0.1531	29.72 dB	12.67 dB	68.6 MB
10	0.1548	29.75 dB	12.69 dB	70.2 MB
5	0.9013	37.85 dB	20.53 dB	68.6 MB
10	0.9023	37.90 dB	20.57 dB	70.2 MB

We also explore scenarios where the latent representations y_1 and y_2 are evenly divided into five and ten groups, respectively, with the results shown in Table 3. It can be observed that when the latent representations are divided into ten groups, the performance increases only slightly. Additionally, the model size significantly increases. This is attributed to the fact that dividing the channels into too many groups results in fewer channels per group, making it challenging to reduce spatial redundancy between pixels.

3 Conclusions

In this paper, we propose two techniques aimed at improving coding performance and speeding up the decoding process. These techniques consist of a dual-branch encoder network and a conditional information module. We employ two independent encoding networks to learn the latent representations of input images at different resolutions, which facilitates the reduction of spatial redundancy in the input images. The high-resolution latent representation primarily captures the global information of the input image, while the low-resolution latent representation predominantly represents its local details. We utilize the high-resolution latent information as side information for the low-resolution latent representation. This approach aids in reducing the spatial redundancy of the low-resolution latent representation, thereby enhancing encoding efficiency. We also employ a parallel channel-wise auto-regressive entropy model [15, 13] for encoding and decoding low-resolution and high-resolution latent representations.

4 Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61474093), Industrial Field Project - Key Industrial Innovation Chain (Group) of Shaanxi Province (2022ZDLGY06-02), the Natural Sciences and Engineering Research Council of Canada (RGPIN-2020-04525), Google Chrome University Research Program, NSERC Discovery Grant RGPIN-2019-04613, DGECR-2019-00120, Alliance Grant ALLRP-552042-2020; CFI John R. Evans Leaders Fund.

References

[1] G. K. Wallace, "The jpeg still picture compression standard," *IEEE Transactions on Consumer Electronics*, vol. 38, no. 1, pp. 18–34, 1992.

[2] A. Skodras, C. Christopoulos, and T. Ebrahimi, "The jpeg 2000 still image compression standard," *IEEE Signal Processing Magazine*, vol. 18, no. 5, pp. 36–58, 2001.

[3] Gary J. Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand, "Overview of the high efficiency video coding (hevc) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[4] Haisheng Fu, Feng Liang, Jianping Lin, Bing Li, Mohammad Akbari, Jie Liang, Guohe Zhang, Dong Liu, Chengjie Tu, and Jingning Han, "Learned image compression with gaussian-laplacian-logistic mixture model and concatenated residual modules," *IEEE Transactions on Image Processing*, vol. 32, pp. 2063–2076, 2023.

[5] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 5718–5727.

[6] Dailan He, Yaoyan Zheng, Baocheng Sun, Yan Wang, and Hongwei Qin, "Checkerboard context model for efficient learned image compression," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 14771–14780.

[7] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto, "Learned image compression with discretized gaussian mixture likelihoods and attention modules," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 7939–7948.

[8] Haisheng Fu, Feng Liang, Jie Liang, Binglin Li, Guohe Zhang, and Jingning Han, "Asymmetric learned image compression with multi-scale residual block, importance

scaling, and post-quantization filtering," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 33, no. 8, pp. 4309–4321, 2023.

[9] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," in *International Conference on Learning Representations*, 2018, pp. 1–23.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021.

[11] Yinhao Zhu, Yang Yang, and Taco Cohen, "Transformer-based transform coding," in *International Conference on Learning Representations*, 2022.

[12] Renjie Zou, Chunfeng Song, and Zhaoxiang Zhang, "The devil is in the details: Window-based attention for image compression," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 17492–17501.

[13] Jinming Liu, Heming Sun, and Jiro Katto, "Learned image compression with mixed transformer-cnn architectures," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 14388–14397.

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

[15] David Minnen and Saurabh Singh, "Channel-wise autoregressive entropy models for learned image compression," in *2020 IEEE International Conference on Image Processing (ICIP)*, 2020, pp. 3339–3343.

[16] 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*, vol. 30, pp. 3179–3191, 2021.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 770–778.

[18] Jiaying Liu, Dong Liu, Wenhan Yang, Sifeng Xia, Xiaoshuai Zhang, and Yuanying Dai, "A comprehensive benchmark for single image compression artifact reduction," *IEEE Transactions on Image Processing*, vol. 29, pp. 7845–7860, 2020.

[19] Y. Hu, W. Yang, and J. Liu, "Coarse-to-fine hyper-prior modeling for learned image compression," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020, vol. 34, pp. 11013–11020.

[20] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto, "Energy compaction-based image compression using convolutional autoencoder," *IEEE Transactions on Multimedia*, vol. 22, no. 4, pp. 860–873, 2020.

[21] Ming Lu and Zhan Ma, "High-efficiency lossy image coding through adaptive neighborhood information aggregation," *arXiv preprint arXiv:2204.11448*, 2022.

[22] Yueyu Hu, Wenhan Yang, Zhan Ma, and Jiaying Liu, "Learning end-to-end lossy image compression: A benchmark," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2021.

[23] G. Bjontegaard, "Calculation of average PSNR differences between RD curves," 2001, VCEG-M33.