2023 IEEE International Conference on Image Processing (ICIP) | 978-1-7281-9835-4/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICIP49359.2023.10222067

CONTENT-ADAPTIVE PARALLEL ENTROPY CODING FOR END-TO-END IMAGE COMPRESSION

*Shujia Li, Dezhao Wang, Zejia Fan, Jiaying Liu**

Wangxuan Institute of Computer Technology, Peking University, Beijing, China

ABSTRACT

State-of-the-art entropy models, *e.g.* autoregressive context models, utilize spatial correlation among latent representations, leading to more accurate entropy estimation. However, this autoregressive design naturally results in serial decoding and the infeasibility of parallelization, which makes the decoding procedure slow and less practical. To address the issue, we propose a Content-Adaptive Parallel Entropy Model (CAPEM) that takes a two-pass context calculation with dynamically generated patterns. Our CAPEM relaxes the strict coding order while the dynamic context mechanism still promotes flexibility in capturing latent dependency. This design greatly improves the parallelism of the context model, leading to higher coding efficiency while maintaining the same rate-distortion performance. We test it on the widely used Kodak and CLIC image datasets. Experimental results show that the proposed model outperforms the recent works with less complexity.

Index Terms— Image Compression, Efficient Entropy Coding, Transformer

1. INTRODUCTION

Image compression is the fundamental technology for image processing, sharing and storage. In recent years, with the rapid advancement of deep learning technology, many efforts have been made to develop learning-based codecs [1, 2, 3, 4, 5]. Unlike traditional compression frameworks, learning-based image compression methods no longer rely on the complex coupling of various manually designed coding tools. Instead, they directly use neural networks for non-linear transform and adaptive entropy coding, which can achieve even better coding performance than the most advanced conventional codecs [6, 7].

Entropy modeling plays a key role in learned compression frameworks. An efficient entropy model can well estimate the probability distribution of the latent representations

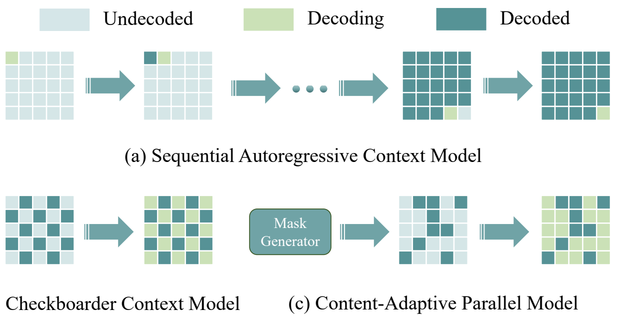

Undecoded
Decoding
Decoded
(a) Sequential Autoregressive Context Model
Mask Generator
(b) Checkboarder Context Model
(c) Content-Adaptive Parallel Model

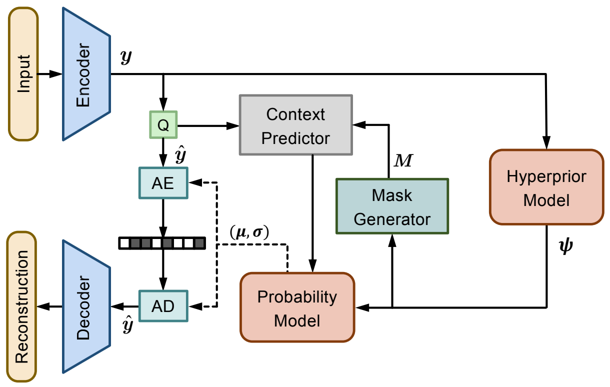
Fig. 1. Three context paradigms. With the proposed two-pass context calculation with dynamically generated patterns, our content-adaptive parallel model makes the best trade-off between efficiency and dynamics.

generated by the neural encoder, which contributes to significant bitrate saving. The factorized model [8] is first proposed for auto-encoder-based frameworks. The entropy is estimated by a variational autoencoder and the latent representation is assumed to be modelled by an independent identically Gaussian distribution. On this basis, hyperprior [2] is further introduced to provide a compact side-information that aims to capture spatial dependencies of the latent representation for conditional probability estimation. However, the learned neural transform cannot be ideal. Therefore, the latent representations are not totally spatially decorrelated, which leads to limited performance in entropy coding.

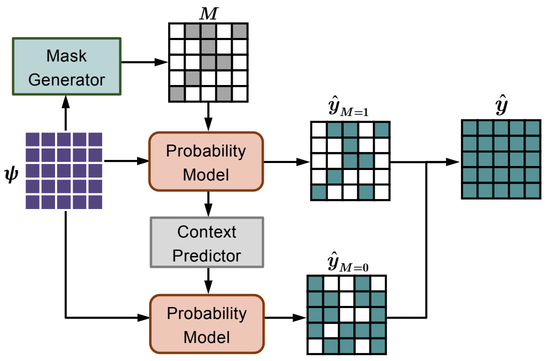
For more accurate probability estimation, researchers propose autoregressive context models [3, 4] to capture spatial dependency in entropy coding. In the autoregressive context model, elements are decoded in a serial way based on the already decoded elements, which is shown in Fig. 1(a). The encoded and decoded parts provide a useful prior for the probability estimation of the part that is being encoded and decoded. Jointly with the hyperprior model, this kind of method can surpass conventional codecs like BPG [7]. However, this autoregressive design naturally results in serial decoding and the infeasibility of parallelization. As a result, the decoding speed of autoregressive context model is extremely slow.

To make the decoding time feasible for practical use, checkerboard context model [9] is proposed to accelerate the serial computing pipeline. As shown in Fig. 1(b), it utilizes

* Corresponding Author. This work was supported by the National Natural Science Foundation of China under Contract No.62172020 a research achievement of Key Laboratory of Science, Techonology and Standard in Press Industry (Key Laboratory of Intelligent Press Media Technology).



(a) Overall Pipeline



(b) Parallelly Entropy Decoding

Fig. 2. The overall structure of the proposed network.

a two-stage decoding strategy. In the first stage, half elements of the latent representation are decoded based on the hyperprior without any context information guidance. The remained parts are then decoded based on these reconstructed elements. As the multi-stage decoding shrinks into only two stages, the deocding process can be parallelized and accelerated. However, the pattern of the element partition map is like a checkerboard, which is used for all kinds of images. It lacks flexible adaption for variable inputs and thus fails to capture the latent characteristics at different locations, *e.g.* various partition ways and orders, which limits the compression performance. As a result, the checkerboard context model can greatly accelerate the decoding procedure. But its compression performance is much inferior to autoregressive context models.

To simultaneously improve rate-distortion performance and decoding efficiency for end-to-end compression frameworks, we propose a Content-Adaptive Parallel entropy Model (CAPEM). In our CAPEM, the decoding is also parted into two stages. Our significant difference is that the partition mask is no longer fixed but dynamically generated according to the image content as shown in Fig. 1(c). It leads to more effective latent relationship modelling with various partition ways and orders. This two-stage flexible design naturally improves parallelism and maintains dynamics, and thus greatly improves decoding efficiency. To further boost the compression performance, we use the Swin Transformer [10, 11] as the transform module. It makes the model attain the information from a global view and is demonstrated outstanding performance in various kinds of visual tasks. The transformation module built using transformers can further enhance feature extraction capabilities and reduce the correlation between features. Our final model can outperform all compared conventional and learned codecs on Kodak [12] and CLIC professional validation datasets [13].

The rest of the paper is organized as follows. Section 2 introduces our proposed framework with content-adaptive parallel entropy model. Details of the framework's architecture and experimental results comparison are shown in Section 3 and concluding remarks are given in Section 4.

2. CONTENT-ADAPTIVE PARALLEL ENTROPY MODEL BASED COMPRESSION

In general, our efforts are put into crafting more strong transforms and more effective entropy models. For the transform, our transform model takes Swin Transformer [10] as the backbone, which effectively introduces the long-term spatial dependencies to obtain more decorrelated latent representations.

For the entropy coding, a content-adaptive parallel entropy model consisting of a mask generator and a context prediction module is built for efficient entropy coding. We will first introduce the general network structure of our proposed framework. Then we will introduce the content-adaptive parallel entropy model in detail.

2.1. Overview of the Proposed Network Structure

Our network structure is shown in Fig. 2(a), which is based on the common auto-encoder-based architecture [2, 3, 4, 11]. The network can be roughly divided into a transform module and an entropy model.

Transform module. The learned image compression paradigm typically includes transform modules such as an encoder g_a and a decoder g_s [2]. When compressing the original image $\boldsymbol{x}$, $g_a(\cdot)$ is used to extract the latent representation $\boldsymbol{y}$ and quantization model $\texttt{round}(\cdot)$ is applied to get the discrete latent representation $\hat{\boldsymbol{y}}$:

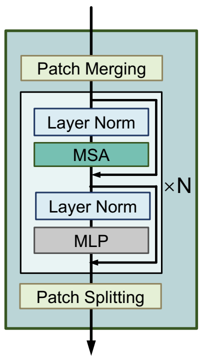
$$\boldsymbol{y} = g_a(\boldsymbol{x}), \hat{\boldsymbol{y}} = \texttt{round}(\boldsymbol{y}). \tag{1}$$

And $g_s(\cdot)$ reconstructs the reconstructed image $\hat{\boldsymbol{x}}$:

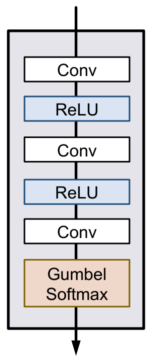
$$\hat{\boldsymbol{x}} = g_s(\hat{\boldsymbol{y}}). \tag{2}$$

Here we use Swin transformer block for g_a and g_s, the structure of our transform module is shown in Fig. 3(a), which consists of a patch merging layer, several Swin Transformer layers and a patch splitting layer.

Entropy model. To entropy-encode $\hat{\boldsymbol{y}}$ into a bitstream, we usually use an entropy estimator to predict the probability distribution of $\hat{\boldsymbol{y}}$, which connects the hyperprior module with context prediction module. The hyperencoder $h_a(\cdot)$ compresses side information into a hyper latent representation z



(a) Swin Transformer Block



(b) Mask Generator

Fig. 3. The detailed architectures of modules.

from $\boldsymbol{y}$ and the hyperdecoder $h_s(\cdot)$ restores the side information $\boldsymbol{\psi}$ from quantized $\hat{\boldsymbol{z}}$:

$$\hat{\boldsymbol{z}} = \texttt{round}(\boldsymbol{z}), \boldsymbol{z} = h_a(\boldsymbol{y}), \boldsymbol{\psi} = h_s(\hat{\boldsymbol{z}}). \tag{3}$$

$\boldsymbol{\psi}$ which contains side information in latent representation $\boldsymbol{y}$, is combined with contextual features of $\hat{\boldsymbol{y}}$ to predict the probability distribution of $\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}$. Assuming that $p_{\boldsymbol{\theta}}(\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}})$ follows a Gaussian distribution $N(\hat{\boldsymbol{y}}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2\boldsymbol{I})$, we can use the reparameterization trick to predict the mean $\boldsymbol{\mu}$ and the scale $\boldsymbol{\sigma}$ like Fig. 2(a):

$$(\boldsymbol{\mu}, \boldsymbol{\sigma}) = g_{pm}(\boldsymbol{\psi}, g_{cp}(\hat{\boldsymbol{y}}_{\boldsymbol{dec}})), \tag{4}$$

where $g_{cp}(\cdot)$ is the context predictor to extract contextual information from decoded $\hat{\boldsymbol{y}}_{\boldsymbol{dec}}$. $g_{pm}(\cdot)$ is the probability model to estimate μ_i and σ_i of each $\hat{y}_i$. Like $p_{\boldsymbol{\theta}}(\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}})$, $p_{\boldsymbol{\delta}}(\hat{\boldsymbol{z}})$ is the probability distribution of $\hat{\boldsymbol{z}}$ predicted by learnable factorized entropy model $\boldsymbol{\delta}$. Specifically, here we use content-adaptive parallel entropy model for flexible and efficient entropy coding.

After estimating the probability distribution $p_{\boldsymbol{\theta}}(\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}})$ and $p_{\boldsymbol{\delta}}(\hat{\boldsymbol{z}})$, $\hat{\boldsymbol{y}}$ and $\hat{\boldsymbol{z}}$ could be entropy-encoded by the arithmetic encoder (AE) into bitstreams:

$$\begin{aligned} \boldsymbol{b_y} &= AE(\hat{\boldsymbol{y}}, p_{\boldsymbol{\theta}}(\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}})), \\ \boldsymbol{b_z} &= AE(\hat{\boldsymbol{z}}, p_{\boldsymbol{\theta}}(\hat{\boldsymbol{z}})). \end{aligned} \tag{5}$$

To obtain a compact bitstream, we optimize the parameters $\boldsymbol{\theta}$ of the entropy estimator to make $p_{\boldsymbol{\theta}}(\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}})$ approximate the true distribution $q(\hat{\boldsymbol{y}})$ as closely as possible. This allows us to optimize the rate-distortion loss via variational inference [8].

2.2. Mask based Entropy Coding

Previous works [3, 4, 5] have demonstrated that the autoregressive context models, which use $\hat{y}_{<i}$ as the context information for $\hat{y}_i$ can achieve excellent compression performance. However, due to its serial design, the decoding procedure is extremely slow, making it infeasible for practical usage. The works in [14, 9] aim to improve decoding efficiency by using a portion of the context information from $\hat{\boldsymbol{y}}$ to predict the distribution of the other portion. Thus, the multi-stage decoding process shrinks to only a two-stage one. Though these context models can accelerate decoding speed, they use fixed context pattern for variable input contents, which does not well organize the coding partition and order and thus limits the compression performance. To improve decoding efficiency while keeping rate-distortion performance, we design a mask generator model $g_{mg}(\cdot)$ to predict a mask conditioned on the input image to group the latent representation in the spatial domain for the two-stage decoding. The mask is of the same size as $\hat{\boldsymbol{y}}$ based on side information $\boldsymbol{\psi}$:

$$\boldsymbol{M} = g_{mg}(\boldsymbol{\psi}). \tag{6}$$

In order to maintain the gradient of generating mask during training, we adopt Gumbel softmax trick [15] in the $g_{mg}(\cdot)$. When estimating the distribution parameters $(\boldsymbol{\mu}, \boldsymbol{\sigma})$ of $\hat{\boldsymbol{y}}$, we first get those about $\hat{\boldsymbol{y}}_{\boldsymbol{M=1}}$:

$$\begin{aligned} (\boldsymbol{\mu_{M=1}}, \boldsymbol{\sigma_{M=1}}) &= g_{pm}(\boldsymbol{\psi}), \\ \hat{\boldsymbol{y}}_{\boldsymbol{M=1}} &= AD(\boldsymbol{b_y}, N(\hat{\boldsymbol{y}}_{\boldsymbol{M=1}}; \boldsymbol{\mu_{M=1}}, \boldsymbol{\sigma_{M=1}}^2\boldsymbol{I})). \end{aligned} \tag{7}$$

After obtaining $\hat{\boldsymbol{y}}_{\boldsymbol{M=1}}$, we use context predictor $g_{cp}(\cdot)$ to extract the contextual information to estimate the distribution of $\hat{\boldsymbol{y}}_{\boldsymbol{M=0}}$:

$$\begin{aligned} (\boldsymbol{\mu_{M=0}}, \boldsymbol{\sigma_{M=0}}) &= g_{pm}(\boldsymbol{\psi}, g_{cp}(\hat{\boldsymbol{y}}_{\boldsymbol{M=1}})), \\ \hat{\boldsymbol{y}}_{\boldsymbol{M=0}} &= AD(\boldsymbol{b_y}, N(\hat{\boldsymbol{y}}_{\boldsymbol{M=0}}; \boldsymbol{\mu_{M=0}}, \boldsymbol{\sigma_{M=0}}^2\boldsymbol{I})), \end{aligned} \tag{8}$$

where $AD(\cdot)$ is the arithmetic decoder. Then, we combine $\hat{\boldsymbol{y}}_{\boldsymbol{M=0}}$ and $\hat{\boldsymbol{y}}_{\boldsymbol{M=1}}$ obtained from two stages to obtain the final decoded $\hat{\boldsymbol{y}}$:

$$\hat{\boldsymbol{y}} = \hat{\boldsymbol{y}}_{\boldsymbol{M=0}} + \hat{\boldsymbol{y}}_{\boldsymbol{M=1}}. \tag{9}$$

The above arithmetic decoding process is lossless and each stage of decoding can be carried out in a parallel manner, which can bring remarkable improvement on the decoding efficiency compared to sequential autoregressive models.

3. EXPERIMENTS

Network Implementation. Specifically, we implement our designed content-adaptive parallel entropy model based on existing end-to-end learning method [11] while preserving the Swin Transformer-based transform modules. We use Swin Transformer blocks interleaved with patch merging blocks to construct the encoder, hyperencoder, hyperdecoder and decoder. In these transform modules, the channel widths are $(128, 192, 256, 320, 192, 192)$ and blocks' depths are $(2, 2, 6, 2, 5, 1)$ in order from main transform with window size 8 to hyper transform with window size 4. The head dimension is 32 for all attention layers in Swin Transformer blocks. Similar to the transform modules, we use a Swin Transformer block to extract contextual information from masked latent code for predicting the remaining unmasked latent code's probability distribution.

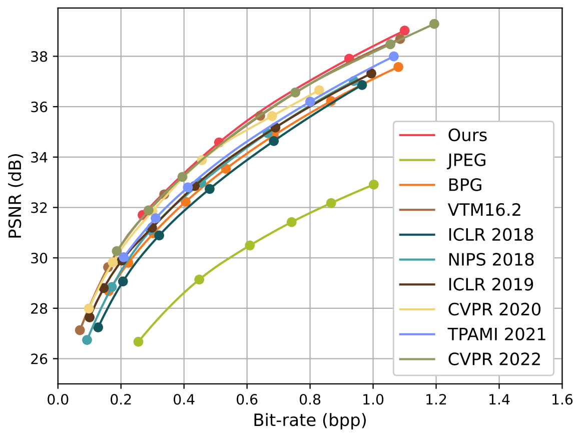


Fig. 4. R-D curves on Kodak.

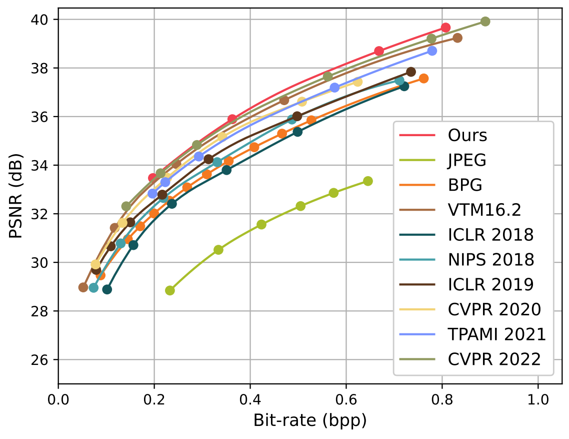


Fig. 5. R-D curves on CLIC professional validation set.

Training Details. We train our CAPEM model on DIV2K [16] dataset. This dataset consists of 800 high-quaily natural images of 2K resolution. For data augmentation, we downsample the 800 images using bilinear interpolation by a factor of 2, and merged them with the original dataset, resulting in a dataset of 1600 images. In each iteration, we randomly crop 256×256 patches from images. The training loss function $L = D + \lambda R$ is designed to improve the Rate-Distortion performance of the compression framework by balancing the distortion D and the bitrate R through adjusting the Lagrangian parameter λ, where Mean Square Error (MSE) is used as the distortion measurement. To cover a wide range of bitrate and distortion, we train four models with $\lambda \in \{3 \times 10^{-3}, 1 \times 10^{-3}, 3 \times 10^{-4}, 2 \times 10^{-4}\}$. We first train a model with the setting of $\lambda = 3 \times 10^{-4}$, and finetune it to three other bitrate points after loading this pretrained weights. During training, we employ a multi-stage training strategy. In stage I, we train transform modules in our CAPEM model for 1.8M iterations with the Adam optimizer [17], whose learning rate is initialized to be 1×10^{-4}. In stage II, we train the whole model for 400k iterations. We still adopt Adam optimizer and set the learning rate to 1×10^{-4}, which is turned down to 1×10^{-5} after 200k iterations.

Evaluation Protocol. We evaluation our model on Kodak image set [12] and the *professional* subset of the CLIC2020 validation dataset [13]. These two commonly used test datasets can be respectively used to evaluate performance of the compression framework on low-resolution and high-resolution images. The performance is mainly measured by rate-distortion performance and compression efficience, where the former is illustrated by R-D curves and BD-rate [18] and the latter is illustrated by decoding time.

Comparison Results. We compare our method with existing learned image compression methods optimized for MSE [4, 5, 14, 19, 9, 20] and conventional image compression methods such as BPG [7] and VVC [6]. We use BPG and VTM-16.2 to compress the images in YUV444 mode, and then calculate Peak Signal to Noise Ratio (PSNR) in RGB mode. The overall results on Kodak and CLIC2020 are shown in Fig. 4 and Fig. 5 respectively.

Table 1. BD-rate results (↓) and decoding time (s) at about 0.6 bpp on CLIC2020 [13]. We set VTM16.2 [6] as the anchor in the calculation.

Codec	Decoding Time (s)	BD-Rate (↓)
NeurIPS 2018 [3]	138.53	16.72%
NeurIPS 2018 + CAPEM	0.77	17.33%
NeurIPS 2018 + Checkerboard [9]	0.76	18.98%
VTM16.2 [21]	0.53	0%
Ours	2.71	**-8.62%**

Moreover, we also compute the BD-rate of each model relative to VTM16.2 and our method can save about 3.19% bitrate on Kodak and 8.62% on CLIC2020 averagely, which outperform existing learned image compression methods. In Tab. 1, we show the decoding time and BD-Rate of several methods. To show efficiency of our Content Adaptive Parallel Emtropy Model (CAPEM), we apply CAPEM and checkerboard entropy model [9] on the sequential autoregressive based work [3]. We can see that our CAPEM has better rate-distortion performance under the condition that the decoding efficiency is not inferior to the checkerboard entropy model. And after using Swin Transformer as transform modules, our model can surpass VVC in rate-distortion performance under when the decoding efficiency is more than ten times faster than the baseline work [3].

4. CONCLUSION

In this paper, we propose a compression framework based on content-adaptive parallel entropy model that takes a two-pass context calculation with dynamically generated patterns. Our model greatly improves the parallelism of the context model while keeping the rate-distortion performance. Experimental results demonstrate that our model obtains better results against previous methods with significantly less complexity.

5. REFERENCES

[1] George Toderici, Sean M O'Malley, Sung Jin Hwang, Damien Vincent, David Minnen, Shumeet Baluja, Michele Covell, and Rahul Sukthankar, "Variable rate image compression with recurrent neural networks," in *Proc. Int'l Conference on Learning Representations*, 2016.

[2] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston, "Variational image compression with a scale hyperprior," in *Proc. Int'l Conference on Learning Representations*, 2018.

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

[4] Jooyoung Lee, Seunghyun Cho, and Seung-Kwon Beack, "Context adaptive entropy model for end-to-end optimized image compression," in *Proc. Int'l Conference on Learning Representations*, 2019.

[5] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto, "Learned image compression with discretized gaussian mixture likelihoods and attention modules," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[6] Benjamin Bross, Jianle Chen, and Shan Liu, "Versatile video coding," *JVET-K1001*, 2018.

[7] Bellard Fabrice, "BPG image format (http://bellard.org/bpg/). accessed: 2021-09," 2018.

[8] Johannes Ballé, Valero Laparra, and Eero P Simoncelli, "End-to-end optimized image compression," in *Proc. Int'l Conference on Learning Representations*, 2017.

[9] Dailan He, Yaoyan Zheng, Baocheng Sun, Yan Wang, and Hongwei Qin, "Checkerboard context model for efficient learned image compression," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[10] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

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

[12] Eastman Kodak, "Kodak lossless true color image suite (photocd pcd0992). [online]. http://r0k.us/graphics/kodak/.," 2013.

[13] "Workshop and challenge on learned image compression," 2020, `http://www.compression.cc`.

[14] David Minnen and Saurabh Singh, "Channel-wise autoregressive entropy models for learned image compression," in *Proc. IEEE Int'l Conference on Image Processing*, 2020.

[15] Eric Jang, Shixiang Gu, and Ben Poole, "Categorical reparameterization with gumbel-softmax," in *Proc. Int'l Conference on Learning Representations*, 2017.

[16] Eirikur Agustsson and Radu Timofte, "NTIRE 2017 challenge on single image super-resolution: Dataset and study," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017.

[17] Diederik Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[18] Gisle Bjontegarrd, "Calculation of average PSNR differences between RD-curves," *VCEG-M33*, 2001.

[19] Yueyu Hu, Wenhan Yang, and Jiaying Liu, "Coarse-to-fine hyper-prior modeling for learned image compression," in *Proc. AAAI Conference on Artificial Intelligence*, 2020.

[20] Dezhao Wang, Wenhan Yang, Yueyu Hu, and Jiaying Liu, "Neural data-dependent transform for learned image compression," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[21] Jianle Chen, Yan Ye, and Seung Hwan Kim, "Versatile video coding (draft 8)," *JVET-Q2002-v3*, 2020.