Context-Based Trit-Plane Coding for Progressive Image Compression

Seungmin Jeon[1], Kwang Pyo Choi[2], Youngo Park[2], Chang-Su Kim[1]*
[1]Korea University, [2]Samsung Electronics
seungminjeon@mcl.korea.ac.kr, {kp5.choi, youngo.park}@samsung.com, changsukim@korea.ac.kr

Abstract

Trit-plane coding enables deep progressive image compression, but it cannot use autoregressive context models. In this paper, we propose the context-based trit-plane coding (CTC) algorithm to achieve progressive compression more compactly. First, we develop the context-based rate reduction module to estimate trit probabilities of latent elements accurately and thus encode the trit-planes compactly. Second, we develop the context-based distortion reduction module to refine partial latent tensors from the trit-planes and improve the reconstructed image quality. Third, we propose a retraining scheme for the decoder to attain better rate-distortion tradeoffs. Extensive experiments show that CTC outperforms the baseline trit-plane codec significantly, e.g. by -14.84% in BD-rate on the Kodak lossless dataset, while increasing the time complexity only marginally. The source codes are available at https://github.com/seungminjeon-github/CTC.

1. Introduction

Image compression is a fundamental problem in both image processing and low-level vision. A lot of traditional codecs have been developed, including standards JPEG [47], JPEG2000 [40], and VVC [11]. Many of these codecs are based on discrete cosine transform or wavelet transform. Using handcrafted modules, they provide decent rate-distortion (RD) results. However, with the rapidly growing usage of image data, it is still necessary to develop advanced image codecs with better RD performance.

Deep learning has been explored with the advance of big data analysis and computational power, and it also has been successfully adopted for image compression. Learning-based codecs have similar structures to traditional ones: they transform an image into latent variables and then encode those variables into a bitstream. They often adopt convolutional neural networks (CNNs) for the transformation. Several innovations have been made to improve RD performance, including differentiable quantization approximations [5, 6], hyperprior [7], context models [20, 32, 33], and prior models [13, 15]. As a result, the deep image codecs are competitive with or even superior to the traditional ones.

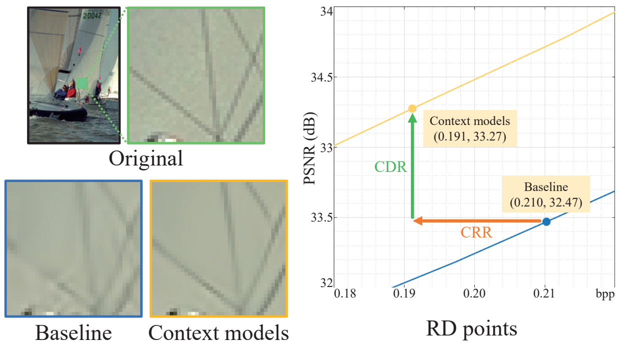


Figure 1. Illustration of the proposed context models: CRR reduces the bitrate, while CDR improves the image quality, as compared with the context-free baseline [27].

It is desirable to compress images progressively in applications where a single bitstream should be used for multiple users with different bandwidths. But, relatively few deep codecs support such progressive compression or scalable coding [35]. Many codecs should train their networks multiple times to achieve compression at as many bitrates [7, 13, 33, 53]. Some codecs support variable-rate coding [15, 51], but they should generate multiple bitstreams for different bitrates. It is more efficient to truncate a single bitstream to satisfy different bitrate requirements. Lu *et al.* [30] and Lee *et al.* [27] are such progressive codecs, based on nested quantization and trit-plane coding, respectively. But, they cannot use existing context models [20,26,32–34], which assume the synchronization of the latent elements, used as contexts, in the encoder and the decoder. Those latent elements are at different states depending on bitrates.

In this paper, we propose the context-based trit-plane coding (CTC) algorithm for progressive image compression, based on novel context models. First, we develop the context-based rate reduction (CRR) module, which entropy-encodes trit-planes more compactly by exploiting already decoded information. Second, we develop the context-based distortion reduction (CDR) module, which refines

*Corresponding author.

partial latent tensors after entropy decoding for higher-quality image reconstruction. Also, we propose a simple yet effective retraining scheme for the decoder to achieve better RD tradeoffs. It is demonstrated that CTC outperforms the existing progressive codecs [27, 30] significantly.

This paper has the following major contributions:

- We propose the *first* context models, CRR and CDR, for deep progressive image compression. As illustrated in Figure 1, CRR reduces the bitrate, while CDR improves the image quality effectively, in comparison with the baseline trit-plane coding [27].
- We develop a decoder retraining scheme, which adapts the decoder to refined latent tensors by CDR to improve the RD performance greatly.
- The proposed CTC algorithm outperforms the state-of-the-art progressive codecs [27, 30] significantly. Relative to [27], CTC yields BD-rates of -14.84% on the Kodak dataset [3], -14.75% on the CLIC validation set [4], and -17.00% on the JPEG-AI testset [1].

2. Related Work

Learning-based codecs: Early learning-based image codecs [19, 23, 44, 45] are based on recurrent neural networks (RNNs), but more codecs [6, 7, 13, 33, 43] employ CNN-based autoencoders [46]. Ballé *et al.* [6] proposed an additive noise model to approximate quantization and trained their network in an end-to-end manner. In [7, 33], hyperprior information was used to estimate the probability distributions of latent elements more accurately. Cheng *et al.* [13] used residual blocks and attention modules in the autoencoder and adopted a Gaussian mixture prior.

Recently, vision transformer [17] or self-attention has been adopted to yield better RD results [24, 38, 53, 54]. Qian *et al.* [38] developed transformer-based hyper-encoder and hyper-decoder. Kim *et al.* [24] decomposed hyperprior parameters to global and local ones. Zou *et al.* [54] used window attention modules in their CNN-based encoder and decoder. Zhu *et al.* [53] adopted the Swin transformer [29] for their encoder, decoder, hyper-encoder and hyper-decoder.

Variable-rate compression: The aforementioned codecs can compress an image at a single rate only. For variable-rate compression, they should be trained multiple times, which is inefficient in both time and memory. In contrast, there are several variable-rate codecs [14, 15, 41, 43, 51]. Theis *et al.* [43] and Choi *et al.* [14] adopted scale parameters for quantization to achieve variable-rate coding. Yang *et al.* [51] adopted the slimmable neural networks [52] and used subsets of network parameters to control bitrates. Cui *et al.* [15] proposed the gain unit for channel-wise bit allocation. Song *et al.* [41] utilized a pixelwise quality map for rate control. These variable-rate codecs support multiple bitrates via single network training, but they still generate separate bitstreams at different bitrates.

Progressive compression: A single bitstream can support multiple bitrates in progressive compression. For example, the traditional JPEG and JPEG2000 have optional progressive modes [40, 47]. Most of learning-based progressive codecs are based on RNNs [19, 23, 44, 45], which support a limited number of quality levels. Also, Cai *et al.* [12] supports only two quality levels with two decoders.

It is more desirable to offer fine granular scalability (FGS) [28, 39]: a single bitstream can be truncated at any point for the decoder to reconstruct an image. Lu *et al.* [30] used nested quantization for FGS. Lee *et al.* [27] proposed trit-plane coding and RD-prioritized transmission of trits. These FGS codecs yield comparable RD curves to conventional deep image codecs.

Context models: As context-based entropy coding techniques such as CABAC [31] are used in traditional codecs [9, 49], context models are also employed in learning-based codecs [20, 26, 32–34]. Minnen *et al.* [33] and Mentzer *et al.* [32] developed autoregressive context models using masked 2D and 3D CNNs, respectively. The autoregressive models exploit spatial contexts serially, demanding high time complexity. Lee *et al.* [26] proposed bit-consuming and bit-free contexts to estimate latent distributions. Minnen *et al.* [34] explored a channelwise autoregressive model with latent residual prediction. He *et al.* [20] developed a checkerboard context model to reduce time complexity.

All these context models can be used for fixed-rate compression only. In contrast, we develop two context models, CRR and CDR, for progressive compression based on trit-plane coding, which improve the RD performance significantly with only a marginal increase of time complexity.

3. Proposed Algorithm

3.1. Trit-Plane Coding

Trit-plane coding was introduced in [27] for deep progressive image compression. Figure 2 shows the framework of the proposed CTC algorithm, which is also based on the trit-plane representation of latent elements. The encoder g_a and the hyper-encoder h_a transform an image $\mathbf{X}$ into a latent tensor $\mathbf{Y}$ and a hyper latent tensor $\mathbf{Z}$ sequentially. Then, using the quantized $\widehat{\mathbf{Z}}$, the hyper-decoder h_s yields $\mathbf{M}$ and $\mathbf{\Sigma}$, representing the mean and standard deviation of $\mathbf{Y}$.

For trit-plane coding, we express the centered and quantized latent tensor $\widehat{\mathbf{Y}} = q(\mathbf{Y} - \mathbf{M})$ in a ternary number system through the trit-plane slicing module: $\widehat{\mathbf{Y}} \in \mathbb{R}^{C\times H\times W}$ is sliced into L trit-planes $\mathbf{T}_l$, $l = 1, \ldots, L$. Each trit-plane is a tensor of the same size as $\widehat{\mathbf{Y}}$. Also, $\mathbf{T}_1$ is the most significant trit-plane (MST), while $\mathbf{T}_L$ is the least significant one (LST). The trit-planes are entropy-encoded into a bitstream progressively from MST to LST. To entropy-encode

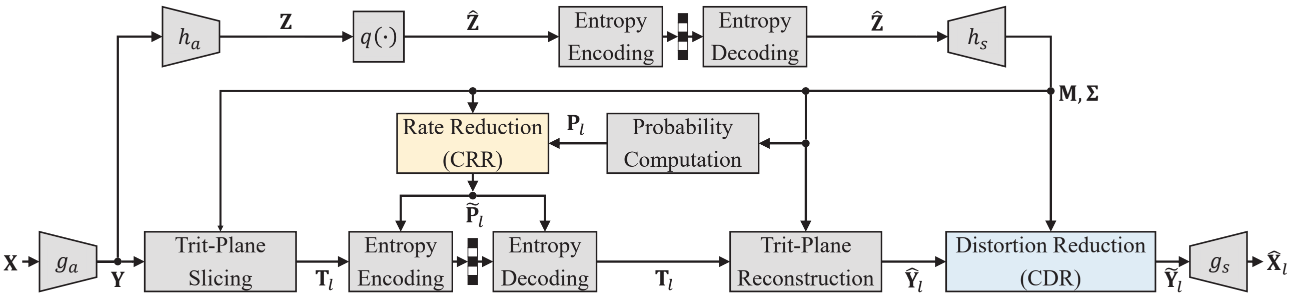


Figure 2. The framework of the proposed CTC algorithm. The context-based rate reduction (CRR) and context-based distortion reduction (CDR) modules are shown in detail in Figure 4.

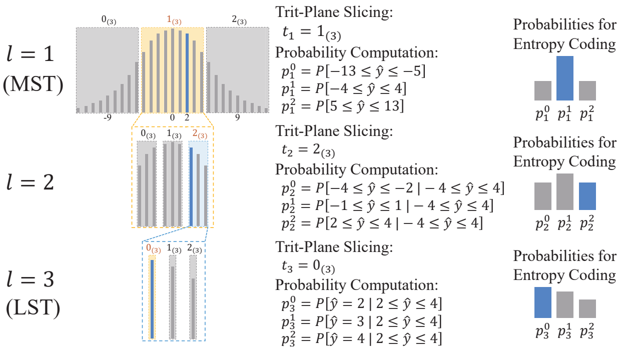


Figure 3. A toy example for trit-plane slicing and probability computation, where an element $\hat{y}$ in $\widehat{\mathbf{Y}}$ equals 2. In trit-plane slicing, we determine where the element belongs among three equal subintervals. In probability computation, we compute the conditional probabilities of each trit. These two steps of trit-plane slicing and probability computation are carried out recursively from MST to LST.

the lth trit-plane $\mathbf{T}_l$, we compute the probability tensor $\mathbf{P}_l$ containing the probabilities that each trit in $\mathbf{T}_l$ equals $0_{(3)}$, $1_{(3)}$, or $2_{(3)}$.[1] Thus, $\mathbf{P}_l \in \mathbb{R}^{3C \times H \times W}$. In Figure 2, the probability computation module estimates $\mathbf{P}_l$ by employing the entropy parameters $\mathbf{M}$, $\mathbf{\Sigma}$ and the already encoded trit-planes $\mathbf{T}_{1:l-1}$. Before the entropy coding, CTC refines $\mathbf{P}_l$ to $\widetilde{\mathbf{P}}_l$ using the CRR module. Then, the trits in $\mathbf{T}_l$ are encoded into a bitstream in the decreasing order of their RD priorities [27]. Figure 3 is a toy example for trit-plane slicing and probability computation.

Conversely, at the decoder side, the trit-planes are entropy-decoded from the bitstream. At any point of the entropy decoding, the image can be reconstructed. Assume that only the first l trit-planes are decoded. Here, l can be a fractional number. For example, if $l = 2.31$, $\mathbf{T}_1$ and $\mathbf{T}_2$ are fully decoded, while 31% of trits in $\mathbf{T}_3$ are decoded. Then, the trit-plane reconstruction module obtains the partial latent tensor $\widehat{\mathbf{Y}}_l$ from the l trit-planes. Specifically, let y be a latent element. Using the available trits, the decoder first identifies the interval $\mathcal{I}$ where y belongs and then reconstructs it to the conditional mean, given by

$$\hat{y}_l = E[y | y \in \mathcal{I}]. \tag{1}$$

Finally, the CDR module reduces distortions in $\widehat{\mathbf{Y}}_l$ to yield $\widetilde{\mathbf{Y}}_l$, and the decoder g_s reconstructs the image $\widehat{\mathbf{X}}_l$ from the refined latent tensor $\widetilde{\mathbf{Y}}_l$.

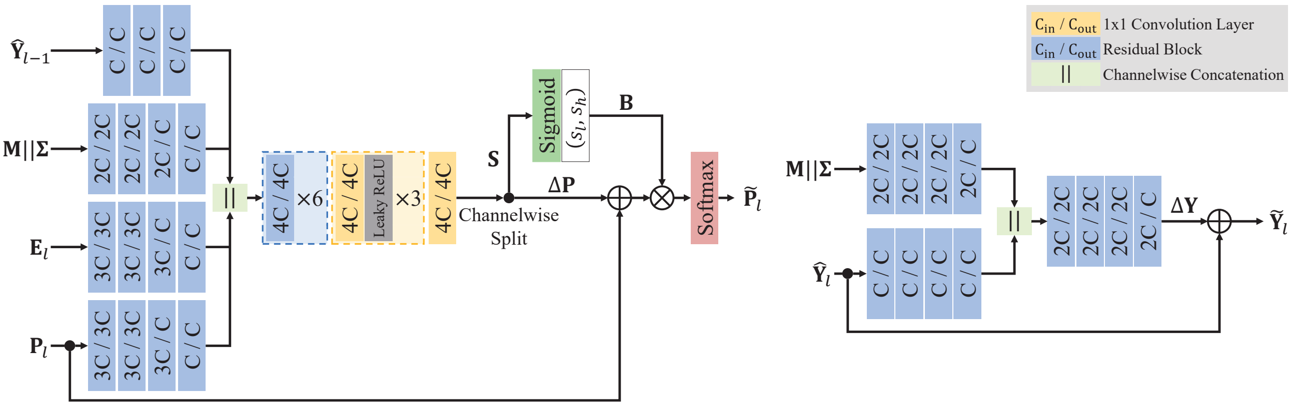
[1] The subscripts (3), indicating the ternary number system, are omitted in the remaining paper for notational convenience.

3.2. Context-Based Rate Reduction

Context models are useful for compressing correlated signals efficiently [9]. In the learning-based codecs, an autoregressive context model [33] predicts the entropy parameters of a latent element using already encoded elements and it improves the RD performance significantly. However, it is impossible to use the autoregressive model for trit-plane coding. The model assumes that $C \times H \times W$ latent elements are coded in the same raster scan order by both the encoder and the decoder. Hence, when trit-planes are only partially reconstructed, the decoder cannot perform the same prediction as the encoder, so the decoding breaks down [27].

We propose the first context models for trit-plane coding. Instead of predicting latent elements in the raster scan order, we predict each trit-plane $\mathbf{T}_l$, $l = 1, \ldots, L$, by exploiting already coded information, including the more significant trit-planes $\mathbf{T}_{1:l-1}$. Note that the probability tensor $\mathbf{P}_l$ is used to encode $\mathbf{T}_l$. We refine the probability estimates in $\mathbf{P}_l$ to yield an updated tensor $\widetilde{\mathbf{P}}_l$ using the CRR module in Figure 4(a). $\widetilde{\mathbf{P}}_l$ requires fewer bits during the entropy coding than $\mathbf{P}_l$ does, improving the RD performance.

To this end, we use already coded information: First, the approximate latent tensor $\widehat{\mathbf{Y}}_{l-1}$, reconstructed from $\mathbf{T}_{1:l-1}$, provides a context. Second, the entropy parameters $\mathbf{M}$ and $\mathbf{\Sigma}$ are concatenated and used as another context. Third, the expected latent tensor $\mathbf{E}_l$ is also used. Assuming that each trit in $\mathbf{T}_l$ equals 0, 1, or 2, the expected value of the corresponding latent element is computed via (1). $\mathbf{E}_l$ contains the three possible values of every latent element, so $\mathbf{E}_l \in \mathbb{R}^{3C \times H \times W}$.



(a) Context-based rate reduction (CRR)

(b) Context-based distortion reduction (CDR)

Figure 4. The architecture of the (a) CRR and (b) CDR modules. Each convolution layer has stride 1 and performs zero padding.

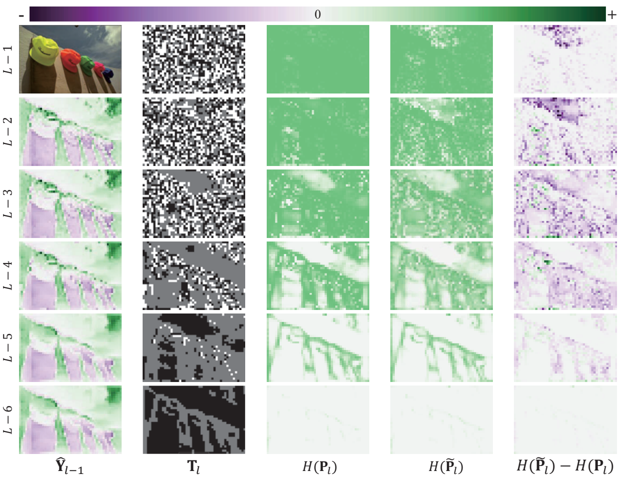


Figure 5. Visualization of $\widehat{\mathbf{Y}}_{l-1}$, $\mathbf{T}_l$, the entropies of $\mathbf{P}_l$ and $\widetilde{\mathbf{P}}_l$, and the residual $H(\widetilde{\mathbf{P}}_l) - H(\mathbf{P}_l)$ in the 141st channel. The top left subfigure is, however, the original image for reference. In $\mathbf{T}_l$, ternary values 0, 1, and 2 are shown in black, gray, and white. In the other cases, green and purple represent positive and negative values, as shown in the top color bar.

In Figure 4(a), CRR extracts features from the input $\mathbf{P}_l$ and the three contexts separately and fuses them lately through residual blocks and convolution layers. The fused tensor has the same spatial resolution as $\mathbf{P}_l$ does, but four times more channels. It is split channelwise into an additive term $\Delta\mathbf{P} \in \mathbb{R}^{3C\times H\times W}$ and a scaling term $\mathbf{S} \in \mathbb{R}^{C\times H\times W}$. First, $\mathbf{S}$ is converted into $\mathbf{B}$ by

$$\mathbf{B} = s_l + (s_h - s_l) \times \text{sigmoid}(\mathbf{S}), \tag{2}$$

whose each element is within (s_l, s_h). Then, $\mathbf{P}_l$ is added to $\Delta\mathbf{P}$, and the sum is modulated by $\mathbf{B}$ to yield an updated probability tensor $\widetilde{\mathbf{P}}_l$. More specifically, let $\{x_0, x_1, x_2\}$ and β be the elements in $(\mathbf{P}_l + \Delta\mathbf{P})$ and $\mathbf{B}$, respectively, corresponding to a trit in $\mathbf{T}_l$. Then, the corresponding updated probabilities $\{\tilde{p}_0, \tilde{p}_1, \tilde{p}_2\}$ in $\widetilde{\mathbf{P}}_l$ are determined using the softmax function,

$$\tilde{p}_i = \frac{e^{\beta x_i}}{\sum_{j=0}^{2} e^{\beta x_j}}, \qquad i = 0, 1, 2. \tag{3}$$

Intuitively, a high β sharpens the probability mass function around the largest input, whereas a low β flattens it. It is proven in Appendix A that the entropy $H(\{\tilde{p}_0, \tilde{p}_1, \tilde{p}_2\})$ is a monotonic decreasing function of β. Thus, to reduce the entropy, we should set a large β. However, the number of required bits is not the ordinary entropy but the cross-entropy

$$\ell_{\text{CRR}} = -\sum_{i=0}^{2} q_i \log_2 \tilde{p}_i, \tag{4}$$

where $\{q_0, q_1, q_2\}$ is the ground-truth one-hot vector for the trit. If the trit corresponds to a highly complicated image region, its probabilities are hard to predict. In such a case, it is beneficial to flatten $\{\tilde{p}_0, \tilde{p}_1, \tilde{p}_2\}$ with a small β and thus to reduce ℓ_{CRR} in (4) on average.

We train CRR to minimize the sum of the cross-entropies in (4) for all trits in $\mathbf{T}_l$. In other words, CRR is learned to modify the input probabilities in $\mathbf{P}_l$ with the additive term $\Delta\mathbf{P}$ and then flatten or sharpen the resulting probabilities with the modulating term $\mathbf{B}$, so the output probabilities in $\widetilde{\mathbf{P}}_l$ minimize the length of the bitstream.

Figure 5 shows that there are spatial redundancies in $\mathbf{Y}$. Hence, neighboring trits in $\mathbf{T}_l$ are also correlated. Even though $\mathbf{T}_l$ for a large l contains more random trits, as indicated by their high entropies in $H(\mathbf{P}_l)$, CRR refines their probability estimates and reduces the entropies in $H(\widetilde{\mathbf{P}}_l)$. The entropy reduction is observed especially in simple regions, such as sky and shadow, as shown in the last column.

It is worth pointing out that CRR can be regarded as a ternary classifier, trained with the cross-entropy loss in (4), that uses the contexts to classify each trit in $\mathbf{T}_l$ into one of the three classes 0, 1, or 2.

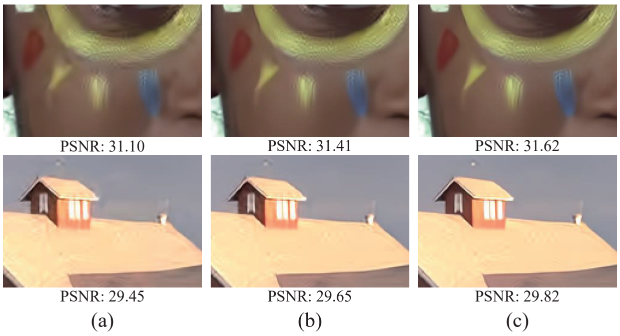


Figure 6. Reconstructed images (a) from partial latent tensors $\widehat{\mathbf{Y}}_{L-2}$, (b) from refined latent tensors $\widetilde{\mathbf{Y}}_{L-2}$ and (c) from $\widetilde{\mathbf{Y}}_{L-2}$ with the retrained decoder.

3.3. Context-Based Distortion Reduction

CRR in Section 3.2, as well as existing context models [20, 32, 33], aims to reduce the required bits for latent elements by predicting their probabilities more accurately. All these context models are used *before* entropy encoding. In contrast, we propose another context model, CDR, that is used *after* entropy decoding. Unlike non-progressive codecs, the proposed algorithm can use a partial latent tensor $\widehat{\mathbf{Y}}_l$, for any $0 < l \leq L$, to reconstruct the image $\widehat{\mathbf{X}}_l$. Thus, after decoding $\widehat{\mathbf{Y}}_l$, which is a truncated approximation of $\mathbf{Y}$, CDR tries to reduce the error $\|\mathbf{Y} - \widehat{\mathbf{Y}}_l\|_F$ using contexts, thereby reducing the image distortion $\|\mathbf{X} - \widehat{\mathbf{X}}_l\|_F$ as well. Here, $\|\cdot\|_F$ denotes the Frobenius norm.

Figure 4(b) shows the architecture of CDR. Using $\mathbf{M}$ and $\mathbf{\Sigma}$ as the contexts, CDR refines the partial latent tensor $\widehat{\mathbf{Y}}_l$ into $\widetilde{\mathbf{Y}}_l$. It regresses the residual $\mathbf{\Delta Y}$ and yields the sum

$$\widetilde{\mathbf{Y}}_l = \widehat{\mathbf{Y}}_l + \mathbf{\Delta Y} \tag{5}$$

as the refined tensor. Note that, different from CRR, CDR does not use $\mathbf{E}_l$ and $\mathbf{P}_l$ as contexts, for they contain probabilistic information about $\mathbf{T}_l$. Since $\mathbf{T}_l$ is already decoded and used to reconstruct $\widehat{\mathbf{Y}}_l$, $\mathbf{E}_l$ and $\mathbf{P}_l$ hardly provide additional information not included in $\widehat{\mathbf{Y}}_l$. Also, note that CDR is a regressor for reducing the distortion, whereas CRR is a classifier for reducing the bitrate.

The CDR module is trained to minimize the loss

$$\ell_{\text{CDR}} = \|\mathbf{Y} - \widetilde{\mathbf{Y}}_l\|_F. \tag{6}$$

For example, Figure 6(a) shows the reconstructed images from partial latent tensors $\widehat{\mathbf{Y}}_{L-2}$, with noticeable compression artifacts. In contrast, Figure 6(b) is the reconstruction from the refined tensors $\widetilde{\mathbf{Y}}_{L-2}$ by CDR, in which the artifacts are alleviated.

3.4. Decoder Retraining

In trit-plane coding, both the encoder and the decoder are trained for a fixed point in the RD curve (usually a high-rate, low-distortion point), and a resultant latent tensor $\mathbf{Y}$ is sliced into trit-planes for progressive compression [27]. We also adopt this strategy to first train the encoder g_a, the hyper-encoder h_a, the decoder g_s, and the hyper-decoder h_s in Figure 2. Then, we obtain $\mathbf{Y}$ and truncate it to various versions $\widehat{\mathbf{Y}}_l$, $0 < l \leq L$. Using these partial tensors $\widehat{\mathbf{Y}}_l$, we train the CRR and CDR modules, respectively, to reduce the required bitrates and the distortions by minimizing the losses in (4) and (6).

In Figure 2, trit-plane slicing and reconstruction are not differentiable, so CRR and CDR, which process trit-planes $\mathbf{T}_l$ and partial tensors $\widehat{\mathbf{Y}}_l$, cannot be trained jointly with g_a, h_a, g_s, and h_s in an end-to-end manner. Hence, we adopt the sequential training scheme.

CDR refines $\widehat{\mathbf{Y}}_l$ into $\widetilde{\mathbf{Y}}_l$, which is used as the new input to the decoder g_s. Thus, we retrain g_s to further improve the quality of the reconstructed image $\widetilde{\mathbf{X}}_l$. Specifically, we generate $\widetilde{\mathbf{Y}}_l$ for various l and retrain the decoder g_s to minimize

$$\ell_{\text{DEC}} = \sum_l w_l \times \|g_s(\widetilde{\mathbf{Y}}_l) - \mathbf{X}\|_F, \tag{7}$$

where w_l is a weighting parameter for each significance level l. The retraining improves the reconstruction quality, as illustrated in Figure 6(c).

4. Experiments

4.1. Implementation and Evaluation

We implement the proposed CTC algorithm based on the Cheng *et al.*'s network [13], composed of residual blocks and attention modules. However, we eliminate the autoregressive model and instead adopt CRR and CDR to exploit contexts. Also, we employ the unimodal Gaussian prior, rather than the Gaussian mixture model in [13], to simplify the latent reconstruction in (1) and the computation of $\mathbf{P}_l$. We use the ANS coder [18] for the entropy coding.

To train CTC, we sample frames from the Vimeo-90K dataset [50] and crop 256×256 patches from each frame as input. We use the Adam optimizer [25] with a batch size of 8 and set a learning rate of 10^{-4} with cosine annealing cycles [21]. First, we train g_a, h_a, g_s, and h_s by approximating the quantizer $q(\cdot)$ in Figure 2 with the additive uniform noise model [6]. Second, we train three sets of CRR and CDR, respectively, for different intervals of l. Third, we retrain the decoder g_s to minimize the loss in (7). More implementation and training details are in Appendix B.

For evaluation, we use the Kodak lossless dataset [3], the CLIC professional validation dataset [4], and the JPEG-AI testset [1]. Kodak consists of 24 images of resolution 512×768 or 768×512, while CLIC and JPEG-AI contain 41 and 16 images of up to 2K resolution. We report bitrates in bits per pixel (bpp) and measure image qualities in PSNR and MS-SSIM [48]. For MS-SSIM, we present decibel scores

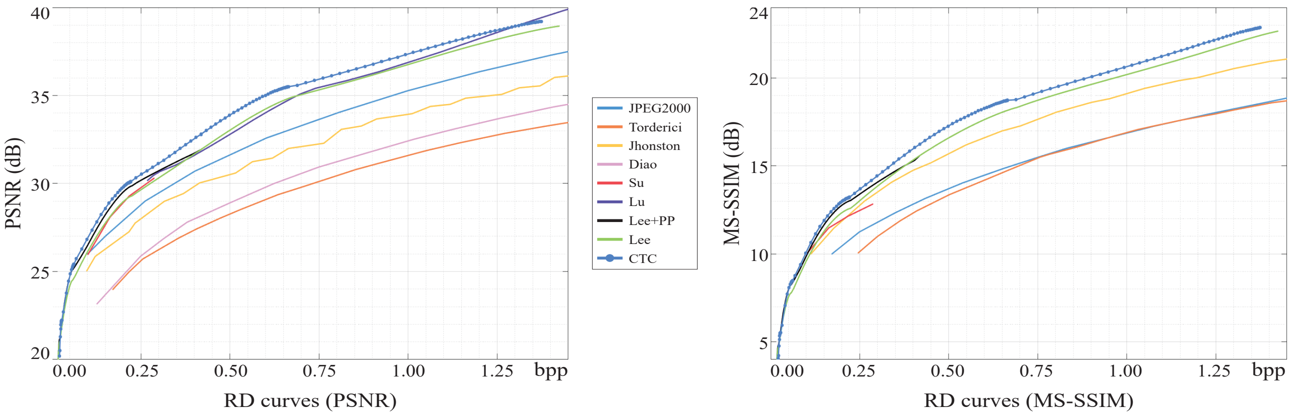


Figure 7. RD curve comparison of the proposed CTC algorithm with existing *progressive* codecs on the Kodak lossless dataset: Torderici *et al*. [45], Jhonston *et al*. [23], Diao *et al*. [16], Su *et al*. [42], Lu *et al*. [30] and Lee *et al*. [27]. '+PP' means that the postprocessing networks are used to improve Lee *et al*. The performance of JPEG2000 is measured in the default non-progressive mode to be used as the same benchmark in both this figure and Figure 8.

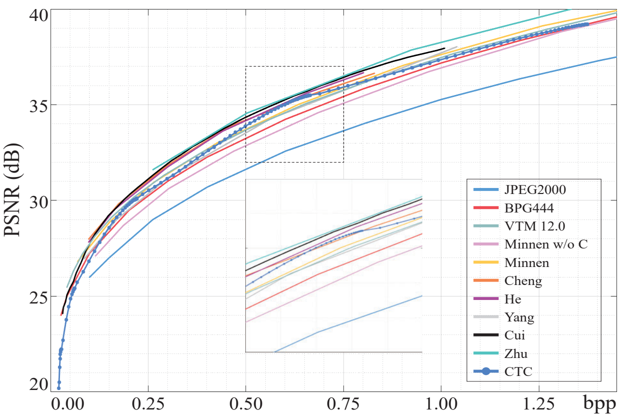


Figure 8. RD curve comparison of CTC with existing *non-progressive* codecs on Kodak: JPEG2000 [2], BPG444 [10], VTM 12.0 [11], Minnen *et al*. [33], Cheng *et al*. [13], He *et al*. [20], Yang *et al*. [51], Cui *et al*. [15], and Zhu *et al*. [53].

by MS-SSIM (dB) $= -10 \cdot \log_{10}(1 - \text{MS-SSIM})$. Also, we compare the compression performances of two algorithms using the BD-rate metric [37].

4.2. Performance Comparison

RD curves: We compare the proposed CTC algorithm with traditional BPG444 [10], VTM 12.0 [11] and learning-based codecs in [13, 15, 16, 20, 23, 27, 30, 33, 42, 45, 51, 53].

Figure 7 compares the RD curves of CTC with those of progressive codecs on the Kodak lossless dataset. CTC outperforms all conventional codecs with meaningful gaps in both PSNR and MS-SSIM at a wide range of bitrates. For example, at 0.5bpp, CTC yields at least 0.8dB better PSNR than the competing codecs Lee *et al*. [27] and Lu *et al*. [30] do. Notice that CTC and these two codecs support FGS. Whereas these codecs do not use any context models, CTC

Table 1. BD-rate performances (%) with respect to Lee *et al*. [27].

		Kodak	CLIC	JPEG-AI
Traditional	JPEG2000 [2]	31.19	48.54	37.46
	BPG444 [10]	−12.16	−1.25	−7.95
	VTM 12.0 [11]	−13.44	−8.75	−14.58
Fixed-rate	Minnen w/o C [33]	−8.61	−0.90	−4.12
	Minnen *et al*. [33]	−16.65	−11.11	−13.92
	Cheng *et al*. [13]	−23.99	−15.99	-
FGS	Lu *et al*. [30]	−0.61	-	2.91
	Lee *et al*. +PP [27]	−6.84	−6.87	−7.19
	CTC	−14.84	−14.75	−17.00

Table 2. Complexity comparison of CTC with Minnen *et al*. [33] and Lee *et al*. [27]. The average encoding and decoding times for a single image in the Kodak lossless dataset are reported.

	# Parameters	Encoding (s)	Decoding (s)
Minnen *et al*. [33]	30.6M	4.01	11.02
Lee *et al*. (+PP) [27]	27.2M (+50M)	1.73	1.40 (+0.10)
CTC	39.9M	1.78	1.55

exploits CRR and CDR and improves the RD curves significantly. On the other hand, Su *et al*. [42] supports a narrow range of bitrates only, while the other learning-based codecs in [16, 23, 45] provide even worse PSNR curves than JPEG2000 [2].

Next, Figure 8 compares CTC with non-progressive codecs: traditional codecs [2, 10, 11], learning-based fixed-rate codecs [13, 20, 33, 53] and variable-rate codecs [15, 51]. 'Minnen w/o C' means the Minnen *et al*.'s network without the context model [33]. Although CTC supports the additional functionality of FGS, it yields a comparable curve to these non-progressive codecs. Especially, around 0.6bpp, CTC provides competitive PSNRs to the existing codecs, including Cui *et al*. [15] and VTM 12.0 [11], which are

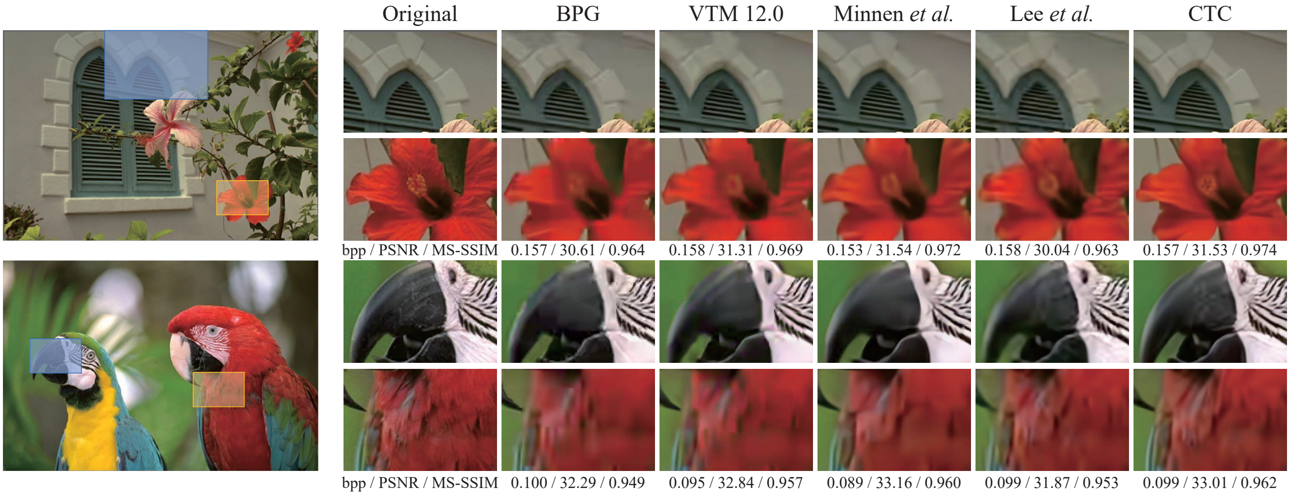


Figure 9. Comparison of reconstructed images by different codecs: BPG [10], VTM 12.0 [11], Minnen *et al*. [33], Lee *et al*. [27] and CTC.

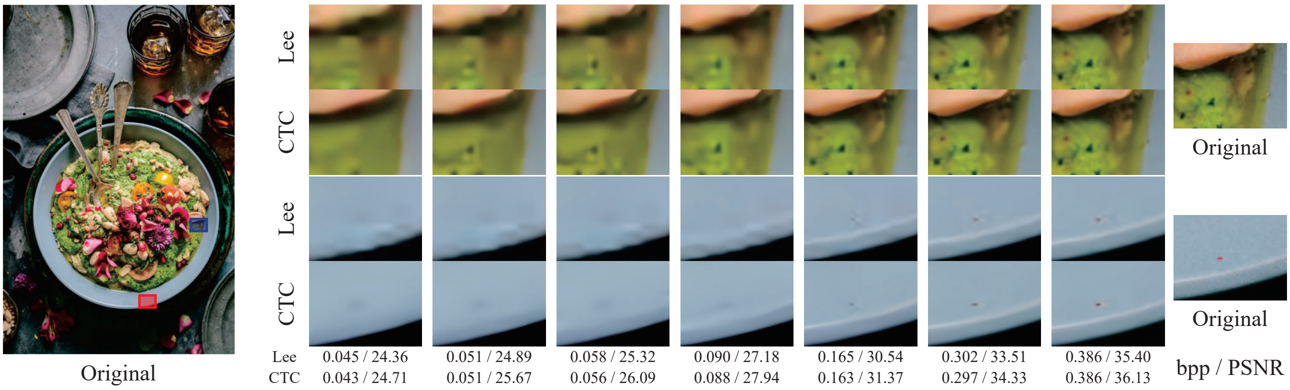


Figure 10. Qualitative comparison of progressive reconstruction results by Lee *et al*. [27] and CTC. The bitrates (bpp) and PSNRs (dB) for the entire image are also listed in the corresponding columns.

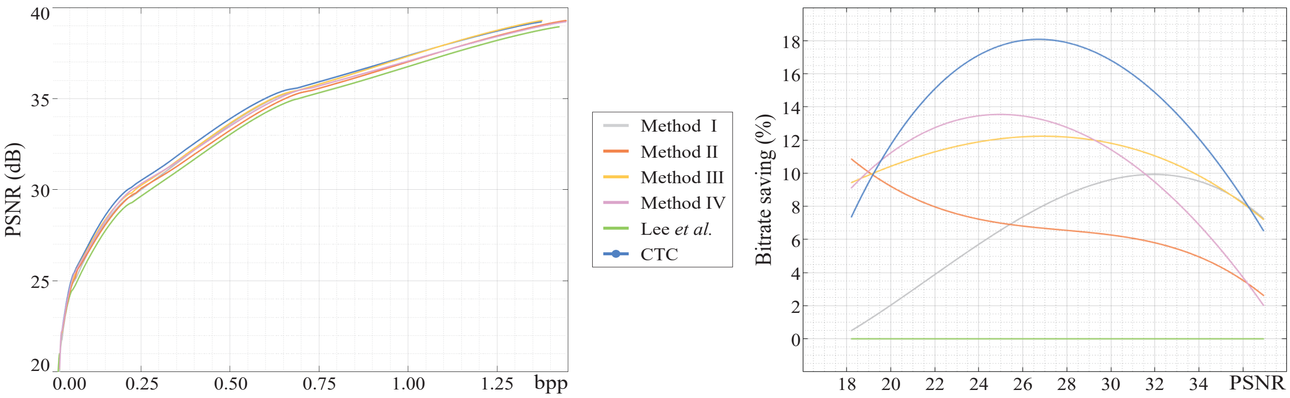
the state-of-the-art variable-rate codecs. Also, CTC outperforms 'Minnen w/o C' [33] and BPG444 [10] at almost every bitrate. More RD curves on other datasets are available in Appendix C.

BD-rates: Table 1 lists the BD-rates relative to Lee *et al*. [27] on the three test datasets. Among the FGS codecs, the proposed CTC provides by far the best results on all datasets. For instance, on JPEG-AI, CTC achieves 17.00% bitrate saving, while Lu *et al*. [30] rather increases the required bitrates. Also, on CLIC, CTC is comparable to Cheng *et al*. [13] and better than VTM 12.0 [11].

Complexities: Table 2 compares the complexities of CTC with those of Minnen *et al*. [33] and Lee *et al*. [27]. Minnen *et al*. is a fixed-rate codec using the autoregressive context model. The Lee *et al*.'s codec supports FGS based on trit-plane coding, but it uses no context model. For Lee *et al*. and CTC, the times are measured for encoding and decoding an entire bitstream.

CTC is much faster than Minnen *et al*., since both CRR and CDR exploit contexts efficiently in parallel using common convolution layers, whereas Minnen *et al*. perform context-based prediction serially. Compared with Lee *et al*., CRR and CDR demand about 12.7M more parameters but increase time complexities only marginally. In other words, CRR and CDR are not only effective for improving the RD performance but also efficient in terms of time complexity. Moreover, in Lee *et al*., the postprocessing (PP) networks are optionally used to improve the reconstruction quality as shown in Figure 8, but they increase the number of parameters by 50M. Without using such PP, CTC outperforms Lee *et al*. significantly.

Qualitative results Figure 9 compares reconstructed images obtained by existing codecs [10, 11, 27, 33] and CTC. Near sharp edges or in textured regions, such as the window and wall patterns, flowers, and feathers, the traditional codecs [10,11] yield blur artifacts. The reconstruction quality of the proposed CTC is better than that of Lee *et al*. [27] and is comparable to that of the Minnen *et al*.'s non-



(a) RD curves (b) Bitrate saving curves

Figure 11. (a) RD curves of the four ablated methods in Table 3 and the baseline codec, Lee *et al*. [27], and (b) the correspoding bitrate saving curves with respect to the baseline.

progressive codec [33].

Figure 10 compares progressive reconstruction results, obtained by Lee *et al*. [27] and CTC. At each column, both trit-plane coding algorithms reconstruct the images $\widehat{\mathbf{X}}_l$ up to the same significance level l. The proposed CTC yields higher RD performances by employing context models and decoder retraining. Consequently, CTC provides a better image quality than Lee *et al*. does.

4.3. Ablation Study

We conduct an ablation study to analyze the three contributions — CRR, CDR, and decoder retraining — of the proposed CTC algorithm as compared with the baseline trit-plane codec, Lee *et al*. [27]. Table 3 lists the BD-rates of four ablated methods relative to the baseline on the Kodak dataset. CRR and CDR in methods I and II improve the RD performances, respectively, by reducing bitrates and improving image qualities. Both CRR and CDR achieve about 7% of bitrate saving. When they are used together, the bitrate saving in method III is as big as 10.93%. Also, the decoder retraining with CDR provides a similar reduction of 10.81%, indicating that the retraining for partial latent tensors $\widehat{\mathbf{Y}}_l$ is also essential in trit-plane coding. By combining the three components, the proposed CTC algorithm achieves a significant bitrate saving of 14.84%.

Figure 11(a) compares the RD curves of the ablated methods in Table 3, and Figure 11(b) plots the bitrate saving percentages in terms of PSNR with respect to the baseline. We see that method I is more effective at a high PSNR range, since trit probabilities can be more accurately predicted using contexts when latent elements are finely reconstructed. On the other hand, method II performs better in a low PSNR range because quantization noise of coarsely reconstructed latent elements can be more easily reduced. The method III exhibits a relatively even bitrate saving in the entire PSNR range. Method IV yields a bitrate saving curve skewed to low PSNRs. Finally, CTC reduces the bitrate requirement significantly, by more than 10%, when PSNR is between 20dB and 35dB. Therefore, the whole bitrate saving is 14.84% as listed in Table 3.

Table 3. Ablation study of CTC: for each ablated method, the BD-rate relative to the baseline, Lee *et al*. [27], is reported.

	CRR	CDR	g_s retraining	BD-rate
Method I	✓	-	-	-6.90%
Method II	-	✓	-	-6.68%
Method III	✓	✓	-	-10.93%
Method IV	-	✓	✓	-10.81%
CTC	✓	✓	✓	$\mathbf{-14.84\%}$

5. Conclusions

We proposed an effective trit-plane codec, called CTC, for progressive image compression using the two context modules: CRR and CDR. Before entropy encoding, CRR updates a probability tensor to compress trit-planes more compactly. After entropy decoding, CDR refines a partial latent tensor to reconstruct a higher-quality image. Both CRR and CDR are based on convolutional layers, so they are efficient in terms of time complexity. Moreover, we developed a decoder retraining scheme, which, combined with CDR, achieves better RD tradeoffs. It was shown that CTC outperforms conventional progressive codecs greatly.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grants funded by the Korea government (MSIT) (No. NRF-2021R1A4A1031864 and No. NRF-2022R1A2B5B03002310), and by IITP grant funded by the Korea government (MSIT) (No. 2021-0-02068, Artificial Intelligence Innovation Hub).

References

[1] *JPEG-AI Test Images*. [Online]. Available: https://jpegai.github.io/test_images. 2, 5

[2] *JPEG2000 Official Software OpenJPEG*. [Online]. Available: https://jpeg.org/jpeg2000/software.html. 6, 11

[3] *Kodak Lossless True Color Image Suite*. [Online]. Available: http://r0k.us/graphics/kodak. 2, 5

[4] *Workshop and Challenge on Learned Image Compression*. 2018. [Online]. Available: http://challenge.compression.cc/tasks/. 2, 5

[5] Eirikur Agustsson, Fabian Mentzer, Michael Tschannen, Lukas Cavigelli, Radu Timofte, Luca Benini, and Luc Van Gool. Soft-to-hard vector quantization for end-to-end learning compressible representations. In *NeurIPS*, 2017. 1

[6] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. End-to-end optimized image compression. In *ICLR*, 2017. 1, 2, 5

[7] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. In *ICLR*, 2018. 1, 2

[8] Jean Bégaint, Fabien Racapé, Simon Feltman, and Akshay Pushparaja. CompressAI: a PyTorch library and evaluation platform for end-to-end compression research. *arXiv preprint arXiv:2011.03029*, 2020. 11

[9] Timothy C. Bell, John G. Cleary, and Ian H. Witten. *Text Compression*. Prenticel Hall, 1990. 2, 3

[10] Fabrice Bellard. *BPG Image Format*. 2014. [Online]. Available: https://bellard.org/bpg. 6, 7, 11

[11] Benjamin Bross, Ye-Kui Wang, Yan Ye, Shan Liu, Jianle Chen, Gary J. Sullivan, and Jens-Rainer Ohm. Overview of the versatile video coding (VVC) standard and its applications. *IEEE Trans. Circuit Syst. Video Technol.*, 31(10):3736–3764, 2021. 1, 6, 7, 11

[12] Chunlei Cai, Li Chen, Xiaoyun Zhang, Guo Lu, and Zhiyong Gao. A novel deep progressive image compression framework. In *Picture Coding Symposium*, pages 1–5, 2019. 2

[13] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. Learned image compression with discretized Gaussian mixture likelihoods and attention modules. In *CVPR*, 2020. 1, 2, 5, 6, 7

[14] Yoojin Choi, Mostafa El-Khamy, and Jungwon Lee. Variable rate deep image compression with a conditional autoencoder. In *ICCV*, 2019. 2

[15] Ze Cui, Jing Wang, Shangyin Gao, Tiansheng Guo, Yihui Feng, and Bo Bai. Asymmetric gained deep image compression with continuous rate adaptation. In *CVPR*, 2021. 1, 2, 6

[16] Enmao Diao, Jie Ding, and Vahid Tarokh. DRASIC: Distributed recurrent autoencoder for scalable image compression. In *Data Compression Conference*, 2020. 6

[17] 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 *ICLR*, 2021. 2

[18] Jarek Duda. Asymmetric numeral systems: Entropy coding combining speed of Huffman coding with compression rate of arithmetic coding. *arXiv preprint arXiv:1311.2540*, 2013. 5

[19] Karol Gregor, Frederic Besse, Danilo J. Rezende, Ivo Danihelka, and Daan Wierstra. Towards conceptual compression. In *NeurIPS*, 2016. 2

[20] Dailan He, Yaoyan Zheng, Baocheng Sun, Yan Wang, and Hongwei Qin. Checkerboard context model for efficient learned image compression. In *CVPR*, 2021. 1, 2, 5, 6, 13

[21] Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E. Hopcroft, and Kilian Q. Weinberger. Snapshot ensembles: Train 1, get m for free. In *ICLR*, 2017. 5

[22] Seungmin Jeon, Jae-Han Lee, and Chang-Su Kim. RD-optimized trit-plane coding of deep compressed image latent tensors. *arXiv preprint arXiv:2203.13467*, 2022. 11, 12

[23] Nick Johnston, Damien Vincent, David Minnen, Michele Covell, Saurabh Singh, Troy Chinen, Sung Jin Hwang, Joel Shor, and George Toderici. Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks. In *CVPR*, 2018. 2, 6

[24] Jun-Hyuk Kim, Byeongho Heo, and Jong-Seok Lee. Joint global and local hierarchical priors for learned image compression. In *CVPR*, 2022. 2

[25] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 5

[26] Jooyoung Lee, Seunghyun Cho, and Seung-Kwon Beack. Context-adaptive entropy model for end-to-end optimized image compression. In *ICLR*, 2019. 1, 2

[27] Jae-Han Lee, Seungmin Jeon, Kwang Pyo Choi, Youngo Park, and Chang-Su Kim. DPICT: Deep progressive image compression using trit-planes. In *CVPR*, 2022. 1, 2, 3, 5, 6, 7, 8, 11, 13

[28] Weiping Li. Overview of fine granularity scalability in MPEG-4 video standard. *IEEE Trans. Circuit Syst. Video Technol.*, 11(3):301–317, 2001. 2

[29] 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 *ICCV*, 2021. 2

[30] Yadong Lu, Yinhao Zhu, Yang Yang, Amir Said, and Taco S. Cohen. Progressive neural image compression with nested quantization and latent ordering. In *ICIP*, 2021. 1, 2, 6, 7

[31] Detlev Marpe, Heiko Schwarz, and Thomas Wiegand. Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard. *IEEE Trans. Circuit Syst. Video Technol.*, 13(7):620–636, 2003. 2

[32] Fabian Mentzer, Eirikur Agustsson, Michael Tschannen, Radu Timofte, and Luc Van Gool. Conditional probability models for deep image compression. In *CVPR*, 2018. 1, 2, 5

[33] David Minnen, Johannes Ballé, and George Toderici. Joint autoregressive and hierarchical priors for learned image compression. In *NeurIPS*, 2018. 1, 2, 3, 5, 6, 7, 8, 12, 13

[34] David Minnen and Saurabh Singh. Channel-wise autoregressive entropy models for learned image compression. In *ICIP*, 2020. 1, 2, 12

[35] Jens-Rainer Ohm. Advances in scalable video coding. *Proc. IEEE*, 93(1):42–56, 2005. 1

[36] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2018. 11

[37] Stephane Pateux. Calculation an excel add-in for computing Bjontegaard metric and its evolution. 6

[38] Yichen Qian, Xiuyu Sun, Ming Lin, Zhiyu Tan, and Rong Jin. Entroformer: A transformer-based entropy model for learned image compression. In *ICLR*, 2022. 2

[39] Amir Said and William A. Pearlman. A new, fast, and efficient image codec based on set partitioning in hierarchical trees. *IEEE Trans. Circuit Syst. Video Technol.*, 6(3):243–250, 1996. 2

[40] Athanassios Skodras, Charilaos Christopoulos, and Touradj Ebrahimi. The JPEG 2000 still image compression standard. *IEEE Signal Process. Mag.*, 18(5):36–58, 2001. 1, 2

[41] Myungseo Song, Jinyoung Choi, and Bohyung Han. Variable-rate deep image compression through spatially-adaptive feature transform. In *ICCV*, 2021. 2

[42] Rige Su, Zhengxue Cheng, Heming Sun, and Jiro Katto. Scalable learned image compression with a recurrent neural networks-based hyperprior. In *ICIP*, 2020. 6

[43] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. Lossy image compression with compressive autoencoders. In *ICLR*, 2017. 2

[44] 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 *ICLR*, 2016. 2

[45] George Toderici, Damien Vincent, Nick Johnston, Sung Jin Hwang, David Minnen, Joel Shor, and Michele Covell. Full resolution image compression with recurrent neural networks. In *CVPR*, 2017. 2, 6

[46] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, 2008. 2

[47] Gregory K. Wallace. The JPEG still picture compression standard. *IEEE Trans. Consum. Electron.*, 38(1):18–34, 1992. 1, 2

[48] Zhou Wang, E. P. Simoncelli, and A. C. Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers*, 2003. 5

[49] Thomas Wiegand, Gary J. Sullivan, Gisle Bjontegaard, and Ajay Luthra. Overview of the H.264/AVC video coding standard. *IEEE Trans. Circuit Syst. Video Technol.*, 13(7):560–576, 2003. 2

[50] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T. Freeman. Video enhancement with task-oriented flow. *Int. J. Comput. Vis.*, 127(8):1106–1125, 2019. 5

[51] Fei Yang, Luis Herranz, Yongmei Cheng, and Mikhail G. Mozerov. Slimmable compressive autoencoders for practical neural image compression. In *CVPR*, 2021. 1, 2, 6

[52] Jiahui Yu, Linjie Yang, Ning Xu, Jianchao Yang, and Thomas Huang. Slimmable neural networks. In *ICLR*, 2018. 2

[53] Yinhao Zhu, Yang Yang, and Taco Cohen. Transformer-based transform coding. In *ICLR*, 2022. 1, 2, 6

[54] Renjie Zou, Chunfeng Song, and Zhaoxiang Zhang. The devil is in the details: Window-based attention for image compression. In *CVPR*, 2022. 2