Learned Image Compression with Generalized Octave Convolution and Cross-Resolution Parameter Estimation

Haisheng Fu, Feng Liang
Highlights

Learned Image Compression with Generalized Octave Convolution and Cross-Resolution Parameter Estimation
Haisheng Fu, Feng Liang

- Learned Image Compression
- Generalized Octave Convolution
- Outperforms the state-of-the-art deep learning-based methods and traditional codecs such as VVC
Learned Image Compression with Generalized Octave Convolution and Cross-Resolution Parameter Estimation

Haisheng Fu\textsuperscript{a}, Feng Liang\textsuperscript{a,}\textsuperscript{*}

\textsuperscript{a}School of Microelectronics, Xi’an Jiaotong University, China

\textbf{ARTICLE INFO}

\textbf{Keywords:}
Multi-Resolution Generalized Octave Convolution Information-Fidelity Loss Cross-Resolution Parameter Estimation (CRPE)

\textbf{ABSTRACT}

Recently, image compression approaches based on deep learning have gradually outperformed existing image compression standards including BPG and VVC intra coding. In particular, the application of the context-adaptive entropy model significantly improves the rate-distortion (R-D) performance, in which hyperpriors and autoregressive models are jointly utilized to effectively capture the spatial redundancy of the latent representations. However, the latent representations still contain some spatial correlations. In addition, these methods based on the context-adaptive entropy model cannot be accelerated in the decoding process by parallel computing devices, e.g. FPGA or GPU. To alleviate these limitations, we propose a learned multi-resolution image compression framework, which exploits the recently developed octave convolutions to factorize the latent representations into the high-resolution (HR) and low-resolution (LR) parts, similar to wavelet transform, which further improves the R-D performance. To speed up the decoding, our scheme does not use context-adaptive entropy model. Instead, we exploit an additional hyper layer including hyper encoder and hyper decoder to further remove the spatial redundancy of the latent representation. Moreover, the cross-resolution parameter estimation (CRPE) is introduced into the proposed framework to enhance the flow of information and further improve the rate-distortion performance. An additional information-fidelity loss is proposed to the total loss function to adjust the contribution of the LR part to the final bit stream. Experimental results show that our method separately reduces the decoding time by approximately 73.35 \% and 93.44 \% compared with that of state-of-the-art learned image compression methods, and the R-D performance is still better than H.266/VVC(4:2:0) and some learning-based methods on both PSNR and MS-SSIM metrics across a wide bit rates.

\section{1. Introduction}

Image compression is a crucial and fundamental topic in the field of multimedia communications. Traditional image compression standards including JPEG [42], JPEG 2000 [37], and BPG (intra-frame encoding of HEVC/H.265) [36] are mainly composed of the following components: linear transform, quantization, and entropy coding.

Recently, image compression approaches based on deep neural networks (DNNs) have achieved competitive results and have gradually outperformed existing image compression standards including BPG and the upcoming VVC intra coding. The end-to-end learned image compression approaches aim to minimize the rate-distortion (R-D) cost. However, it is difficult to establish accurate and differentiable estimates of the entropy. One of the most common schemes is to assume that the symbol streams are independent and identically distributed (i.i.d.), and then use parametric models to estimate the distribution of the symbols. In [38, 1, 33], different end-to-end learned image compression frameworks based on DNNs were presented, and all the symbols are assumed to be i.i.d.. Since the entropy models are not established accurately enough, the performances of these methods are limited. To establish a more accurate entropy model, some image compression approaches based on the hyperprior model and joint model were proposed. In [31], the autoregressive information was extracted by 2D $5 \times 5$ shielding convolution for probability estimation, which is the first approach to outperform BPG (4:4:4) in PSNR. In [11], Cheng et al. proposed the discretized Gaussian Mixture Likelihoods (GMMs) module to parameterize the distributions of the latent representations, which can exploit a more efficient and flexible entropy model to capture spatial dependencies of the latent representations.

Many learned image compression methods exploit information enhancement methods to further improve the R-D performance. In [6], the generalized divisive normalization (GDN) operator was introduced into the transformation function for joint nonlinearity and spatial adaptability, which was proven to be very suitable for natural image compression. In [11], the simplified attention module is introduced to improve the R-D performance, which is easier to obtain global information and reduce time complexity. In [32], two enhanced modules including channel-conditioning module and latent residual prediction module are presented, which lead to network architectures with better R-D performance than some existing context-adaptive learned schemes while minimizing serial processing.

Different approaches have been proposed to design an efficient and effective network structure to improve the R-D performance of image compression. The designed structure can extract more accurate and flexible latent representations of the input image and obtain high-quality decompressed images from the quantized compressed feature. For example, in [41, 39, 24], the recurrent neural networks were presented to...
compress the residual information recursively, which mainly relied on binary representation at each iteration to achieve scalable coding. In [33, 34, 2], the generative models were exploited to learn the distribution of input images, which can achieve better subjective quality at extremely low bit rates by adversarial training.

Most DNN-based image compression schemes [1, 6, 16, 17, 30, 5, 41, 38, 33] are based on the autoencoder framework to extract a low-dimensional latent representation of the input image. In [7], Ballé et al. proposed a two-layer approach, which introduces the hyper layer on top of the autoencoder layer. The hyper layer (including a hyper encoder and a hyper decoder) can capture spatial dependencies among the elements of the latent representation of the autoencoder layer. Based on [7], in [22], a coarse-to-fine framework with multi-layers was proposed to effectively remove spatial redundancy, which achieved better performance than [7]. Also, the information aggregation reconstruction sub-network is designed for reconstruction quality, which combines the output of the main decoder and the two hyper decoders to generate the final reconstructed image. The information aggregation module will significantly increase network complexity and computational complexity. In this paper, we just adopt the multi-layer hyper-prior method to our framework. And other modules are introduced to take the place of the information aggregation reconstruction sub-network in [22]. Based on [22], our new framework can achieve better image compression performance and reduces the encoding and decoding time.

Although the approaches based on DNNs have achieved promising results, there are still some rooms for improvement. First, all latent representations have the same resolution, but we know from wavelet-based image compression that multi-resolution schemes can have better R-D performance for natural images [37]. Second, in these context-adaptive entropy model approaches, the encoding of an coefficient depends on the previously decoded neighbor representations. This sequential processing cannot be accelerated by parallel computing devices, e.g. FPGA or GPU. The decoding processing will take a particularly long time.

To address these two question, in this paper, we propose an effective and efficient framework, which can improve both the parallel decoding and R-D performance of image compression. We can achieve a good trade-off between rate-distortion performance and decoding time. The contributions of the proposed method can be summarized as follows:

- We combine the multi-layer hyper-prior method [22] and the GoConv modules [10] to form a new network architecture, which can further reduce the spatial correlations among the latent representation and improve the R-D performance of image compression.

- More advanced cross-resolution parameter estimation (CRPE) modules are introduced in our scheme to improve the performance, which contains two types. The first type is incorporated in between high and low resolutions, which mainly improves the flow of information. The second type of the CRPE module is in different layers, which can combine the low-resolution (LR) parts of different layers. It is easier to obtain the global information for better encoding and decoding. The signal flow of most image compression frameworks is unidirectional. These two CRPE modules can improve the flow of information, in which the decoded information is provided as side information to the to-be-decoded elements.

- To adjust the proportion of the LR part in the output bit stream, an additional information-fidelity loss is proposed and added to the total loss function. Without this information-fidelity loss, the LR part will only occupy a very small proportion of the whole bit stream, which means the LR part is not fully utilized.

- Our method separately reduces the decoding time by approximately 73.35 % and 93.44 % compared with that of state-of-the-art learned image compression methods on Kodak dataset.

Since we do not use element-wise context model, the proposed method can be decoded in parallel. Thanks to the contributions above, our scheme still achieves comparable performance in both PSNR and MS-SSIM quality metrics across a wide range of bit rates, in comparison with the traditional image codecs including VVC (4:2:0) and some recent learned image compression methods on both Kodak dataset and T ecnick dataset.

The remainder of the paper is organized as follows. In Section 2, We briefly overview some relevant methods. In Section 3, we propose our multi-layer hyper-prior and cross-resolution parameter estimation image compression method. In Sec. 4, we compare the proposed scheme with some state-of-the-art learned compression methods and traditional methods. We also conduct a series of ablation experiments to verify the performance of our proposed scheme. The conclusions are given in Sec. 5.

2. Related work

2.1. Learned image compression

Recently, the learned image compression methods mostly are based on autoencoders which have achieved promising results and gradually outperform classical image codecs including VVC. In [7], a hyperprior network is first proposed to estimate the model parameters of the latent representation, which can further capture the spatial correlation. It achieves comparable performance with BPG (4:4:4) codec in terms of the PSNR metric. To further reduce the redundancy of the latent representation, the context model is proposed in learned image compression framework. In [31], an autoregressive mask convolution is further employed in order to get context-adaptive entropy coding. It is the first method which outperforms BPG (4:4:4) in PSNR. In [25], two types of contexts, bit-consuming contexts and bit-free contexts, are further developed. In [11] and [18], the performance is further improved by introducing advanced entropy coding models, such as GMM and GLLMM. Since these methods [31, 25, 11, 18] utilize the context model to improve the R-D
2.2. Octave Convolution

In the classical convolution, all input and output feature vectors have the same spatial resolution. There is some spatial redundancy in low-resolution information, which is not efficient on both memory and computation cost. To alleviate this issue, the octave convolution was first proposed in [10], which factorizes the feature vectors into the high-resolution (HR) and low-resolution (LR) components, similar to wavelet transform [37]. The octave convolution has been successfully employed in the field of many computer vision tasks [10, 44, 4, 3, 27].

In [3], Akbari et al. proposed the generalized octave convolution (GoConv) and applied it to learned image compression. To improve the image compression performance, the context-adaptive entropy model is also used in [3]. Therefore although the performance of [3] is pretty good, the decoding complexity is too expensive owing to the entropy model. In [27], Lin et al. simplify the framework by removing the context model from the network, at the price of reduced performance. The Lagrangian parameter is introduced in the GoConv modules to develop a variable-rate scheme, in which the bitrate range can be controlled more precisely. As in [7], both [27] and [3] are based on two-layer autoencoder models. However, these two methods just introduce the GoConv to take the place of the classical convolution, and they don’t consider the effect between high-resolution (HR) and low-resolution (LR) components.

3. The Proposed Image Compression Framework

In this section, we first describe the overall architecture of the proposed framework, and then explain the details of major components, including the encoder/decoder networks, generalized octave convolution, the cross-resolution parameter estimation (CRPE) modules, and loss functions.

3.1. The Overall Architecture

The proposed framework is described in Fig. 1. It is mainly composed of the core autoencoder, two layers of hyper-prior subnetworks, and three cross-resolution parameter estimation (CRPE) modules. The core autoencoder includes...
Encoder ($g_e$) and Decoder ($g_d$). The first layer of hyperprior subnetwork consists hyper Encoder ($h_1$) and hyper decoder ($h_2$). The second layer of hyper-prior subnetwork includes hyper encoder ($f_2$) and hyper decoder ($f_d$). Three cross-resolution parameter estimation (CRPE) modules contain CRPE module ($E_1$), CRPE ($E_2$), and CRPE ($E_3$). The Encoder ($g_e$) is used to learn a quantized latent representation of the input image, while the entropy subnetwork ($h_1$) and ($f_2$) are expected to learn a probabilistic model over the quantized latent representations of the previous layer, which is utilized for entropy coding or sent to an information module. The notations are summarized in the Table 1.

In the encoding processing, the input image $x$ is transformed into the HR latent representation $y^H$ and the LR latent representation $y^L$ by the Encoder network ($g_e$), which aims to reduce the pixel-wise redundancy of the input image. The details are explained in Sec. IIB. The GDN module in [11] is adopted as the activation in the Encoder network ($g_e$). The $y^L$ and $y^H$ be quantized as $\tilde{y}^L$ and $\tilde{y}^H$. To further reduce the redundancy within the latent representations, $\tilde{y}^L$ and $\tilde{y}^H$ are then further analyzed and sent to the hyper layers to further squeeze out the redundancy. The $\tilde{y}^L$ and $\tilde{y}^H$ will be obtained by the hyper encoder ($h_1$). Last, $\tilde{y}^L$ and $\tilde{y}^H$ will be sent to the second hyper encoder ($f_2$) to obtain the $z^L$ and $z^H$, which will be quantized as $\tilde{z}^L$ and $\tilde{z}^H$ respectively. $\tilde{z}^L$ and $\tilde{z}^H$ will be encoded by factorized entropy encoder without building a prior probability model. $\tilde{y}^L$, $\tilde{y}^H$, $\tilde{z}^L$, $\tilde{z}^H$ are all coded by arithmetic coding. We estimate the probability distribution of $\tilde{y}_1^L$, $\tilde{y}_1^H$, $\tilde{z}_1^L$ and $\tilde{z}_1^H$ with the prediction model. The distribution of each element in $\tilde{y}_1^L$, $\tilde{y}_1^H$, $\tilde{z}_1^L$ and $\tilde{z}_1^H$ is assumed to be a Gaussian distribution with mean $\mu$ and variance $\sigma^2$.

In the decoding processing, as in [7], the quantized $\tilde{y}^L$ and $\tilde{z}^H$ will be obtained by the factorized entropy decoder without building a prior probability model. The quantized $\tilde{z}^L$ and $\tilde{z}^H$ will be sent to the second hyper decoder network to establish the distribution of each element in $\tilde{y}_1^L$. The decoded $\tilde{y}_1^L$ information will be added to the first CRPE module ($E_1$) to estimate the the distribution of each element in $\tilde{y}_1^H$. The decoded $\tilde{y}_1^H$ and $\tilde{y}_1^L$ will be sent to the first hyper decoder $h_d$ to obtain $\psi_h$. The $\psi_h$ and the decoded $\tilde{y}_1^L$ will be sent to the CRPE ($E_2$) to estimate the distribution of each element in $\tilde{y}_1^L$. The decoded $\tilde{y}_1^L$ and $\tilde{y}_1^H$ will be sent to CRPE ($E_3$) to estimate the distribution of each element in $\tilde{y}_1^H$. Finally, the decoded $\tilde{y}_1^H$ and $\tilde{y}_1^L$ will be sent to the decoder ($g_d$) to obtain the reconstructed image. The inverse GDN is adopted in the Decoder network ($g_d$). Note that the latent representations (e.g. $\tilde{y}_1^H$, $\tilde{y}_1^L$) are quantized, which cause some errors during this processing. To address this problem, uniform noises are added to the latent representations in the training stage. The uniform quantization (i.e., round function in this work) is applied to the latent representations during the test. As illustrated in Fig. 2, the quantized HR and LR latent representations are entropy-coded using two separate arithmetic encoder and decoder.

In the ablation experiments in Sec. III, we will explore...
the performance gains of the additional hyper layer and the CRPE modules in our scheme. Since the additional layer does not adopt the context model, the decoding complexity will not increase too much compared to the two-hyper prior model.

3.2. Network Architecture

Octave convolution is similar to the wavelet transform [37], since it can factorize each channel into HR and LR components. The generalized octave convolution (GoConv) and generalized octave transposed-convolution (GoTConv) in [3, 27] are shown in Fig. 3. The strided convolutions are introduced in [27] to replace the sub-sampling operation in the original octave convolution in [10].

The encoder and decoder network structures of the proposed scheme are described in Fig. 2. The size of the input image \( X \) is denoted as \( W \times H \times 3 \), where \( W \) and \( H \) represent the width and height of the image, respectively. In order to make the network easier to train and converge, the input image is normalized to \([-1, 1]\) by calculating \((- \frac{X}{127.5} - 1.0)\). The analysis transforms and synthesis transforms in the autoencoder layers and the two hyper layers have symmetric networks, apart from using convolutional and de-convolutional filters, respectively.

The GoConv and GoTConv architectures are introduced to the Encoder \( (g_e) \) and Decoder \( (g_d) \). GDN/IGDN transforms are respectively adopted in GoConv and GoTConv in the first-layer encoder and decoder, while leaky ReLU is utilized in the second and third hyper autoencoder and the CRPE modules.

The encoder \( (g_e) \) of the first layer consists of four GoConv modules, each is followed by down-sampling with a factor of 2. The hyper encoder \( (h_e) \) of the second layer consists of three GoConv modules. The down-sampling factors are set to 1, 2, 2, separately. The hyper encoder \( (f_e) \) of the third layer is comprised of three GoConv modules, each is followed by a convolution with a stride of 1. The synthesis transform mirrors the analysis transform of the three layers. The last convolution layer with three filters is expected to generate the decompressed image in RGB space.

3.3. The Cross-Resolution Parameter Estimation (CRPE)

The CRPE modules including \( E_1, E_2, E_3 \) are introduced in our scheme to improve the performance, which contain two types, shown in Fig. 1 and Fig. 2. The first type is incorporated in high and low resolutions modules. Our first CRPE modules aim to improve the image compression. The second type of the CRPE modules is in different layers, which can combine the LR parts of different layers. The second type of the CRPE modules are easier to obtain the global information of different layers, which can provide more accurate information for encoding and decoding. These two types of CRPE modules can improve the flow of information, in which the decoded information is provided as side information to the to-be-decoded elements.

Each CRPE consists of several up-sampling convolution layers with a stride of 2 and several normal convolution modules with a stride of 1, where the up-sampling multipliers are determined by the ratio of the width or height of the input two tensors. For example, if the width and height of one input tensor is four times the width and height of another tensor, two up-sampling layers with a stride of 2 are required.

The R-D performance of the learned image compression methods without using the context model will drop significantly. The element being decoded can make full use of the previously decoded neighbor representations. The signal flow of most image compression frameworks is unidirectional. The proposed CRPE can realize the fusion of different information. The CRPE modules can provide decoded information as additional information to undecoded elements, so that element decoding can utilize more global information instead of local information.

We mainly consider the following two aspects in the process of designing the CRPE to further improve the performance of image compression. First, we decode the LR latent representation instead of decoding low bitrate and high bitrate elements simultaneously in parallel. The LR latent representation is first decoded, and then the decoded LR information is provided to the HR information for decoding. Second, the first hyper decoder can exploit the decoded information of the second hyper layer to decode.
Table 2
The average Encoding time (s), Decoding time(s) and model size (MB) on Kodak and Tecnick datasets.

| Dataset | Scheme | Bit Rate | Encoding Time (s) | Decoding Time (s) | Model size (MB) |
|---------|--------|----------|-------------------|------------------|-----------------|
| Kodak   | VVC [15] | Low      | 402.27s           | 0.60s            | None            |
|         | Lee2019 [25] | Low      | 10.721s           | 37.88s           | 123.8MB         |
|         | Hu2021 [23] | Low      | 35.72s            | 77.33s           | 84.6MB          |
|         | Cheng2020 [11] | Low | 20.89s            | 22.14s           | 50.8MB          |
|         | Chen2021 [9] | Low      | 402.26s           | 2405.14s         | 300.99MB        |
|         | Akbari2021 [3] | Low    | 73.46s            | 75.14s           | 74.67MB         |
|         | Ours    | Low      | 6.87s             | 5.90s            | 77.7MB          |
| Tecnick | VVC [15] | High     | 760.81s           | 0.81s            | None            |
|         | Lee2019 [25] | High    | 21.74s            | 70.44s           | 292.6MB         |
|         | Hu2021 [23] | High    | 70.75s            | 207.77s          | 290.9MB         |
|         | Cheng2020 [11] | High | 90.91s            | 93.28s           | 175.18MB        |
|         | Chen2021 [9] | High    | 360.46s           | 10653.14s        | 300.99MB        |
|         | Akbari2021 [3] | High    | 186.38s           | 190.36s          | 218.47MB        |
|         | Ours    | High     | 7.21s             | 6.11s            | 282.0MB         |

3.4. Loss Function

The objective function for training is composed of two terms: the rate $R$, which is the expected length of the bitstream, and distortion $D$, which is the expected error between the input and decompressed images. The tradeoff between the rate and distortion is determined by a Lagrange multiplier denoted by $\lambda$. The R-D optimization problem is then defined as follows:

$$L = R + \lambda D$$

$$R = R^H + R^L$$

$$D = E_{x \sim P_x}[d(x, \hat{x})]$$

(1)

where $P_x$ represents the unknown distribution of the natural images. $D$ indicates the distortion metric such as the mean squared error (MSE) or MS-SSIM. $R^H$ and $R^L$ are the rates corresponding to the HR and LR parts of the output bitstream.

To increase the LF part in the whole bitstream during the training of the multi-layer network, an additional information-fidelity loss is presented. If the information-fidelity loss function is not adopted in the proposed framework, the LR part will only take a very small portion in the final bitstream, which does not take full advantage of the multi-resolution feature. This problem can be effectively addressed by introducing the information-fidelity loss function. This loss term encourages the LR module to maintain the LR critical information during training, which is formulated as:

$$L_{if} = \lambda_1 \| F(y^L, \theta) - y^L \|^2 + \lambda_2 \| F(y^L_1, \omega) - y^L_1 \|^2$$

(2)

The function $F$ with trainable parameter $\theta$ and $\omega$ is one convolutional layer with no non-linear activation. By including $y^L$ and $y^L_1$ in the cost function, more LR bits can be produced by the optimized networks. The parameters $\lambda_1$ and $\lambda_2$ are scaling factors. Ablation experiments will verify the role of the information-fidelity loss. Therefore, the final loss function is as follows:

$$L = R + \lambda D + L_{if}$$

(3)

4. Experiments

In this section, we evaluate the performance of the proposed method on the Kodak PhotoCD dataset [13] and Tecnick dataset [14]. The Kodak PhotoCD dataset is composed
of 24 lossless PNG format images. And the resolution of each test image mainly contains two types: 768 × 512 and 512 × 768. The Tecnick dataset contains 100 lossless PNG images, in which each test image have a resolution of 1200 × 1200. We compare with some recent learning compression methods, including Zhu2022 [45], Ma2022 [29], Guo2022 [19], Ho2021 [21], Chen2021 [9], Cheng2020 [11], Lee2019 [25], Lin2020 [27], Hu2021 [23], Li2020 [26], VVC (4:4:4) [15], VVC (4:2:0) [15], BPG (4:4:4) [36], JPEG2000, WebP, JPEG.

The origin MS-SSIM values are mapped to decibels $-10 \log_{10}(1 - MS-SSIM)$ to better distinguish the performance difference.

### 4.1. Training Set

The CLIC dataset [12] and LIU4K dataset [28] are used to train our network. All the training images are resized to 2000 × 2000 to train our models. Data augmentation approaches including rotation and scaling were introduced to randomly extract 81,650 patches with 384 × 384. The patches are saved as a lossless PNG format.

### 4.2. Parameter Settings

We optimized the proposed models using two evaluation metrics including mean square error (MSE) and MS-SSIM [43]. When optimized for MSE metric, the parameter $\lambda$ is
selected from the set \{0.003, 0.008, 0.02, 0.04, 0.07\}, each trains a model for a different bit rate. The number of filters \(N\) is set to 256 for two lower bit rates, and is set to 448 for the three higher bit rates. When the MS-SSIM metric is used, the parameter \(\lambda\) is in the set \{0.003, 0.008, 0.0015, 0.02, 0.04\}. The number of filters \(N\) is set to 256 for two lower bit rates, and is set to 448 for the three higher bit rates. In the GoConv part, half of the latent representations have low resolutions, and the other half have high resolutions.

The network training process can be divided into three stages. In the first stage, we pretrain the main encoder \(g_e\) and decoder \(g_d\) to obtain a good reconstructed images. The parameters of hyper encoders and decoders are not involved in training. In the second stage, we randomly initialize the hyper encoder \(f_e\) and the decoder \(f_d\). The parameters of the \(f_e\) and \(f_d\) added to training parameters to jointly train the model. In third stage, we randomly initialize the hyper encoder \(h_e\) and the decoder \(h_d\), the whole model is trained in an end-to-end manner. Each model is trained up to 150,000 iterations for each \(\lambda\) to obtain stable performance. The Adam solver with a batch size of 8 is adopted and the learning rate is set to 0.0004 during the training.

4.3. Encoding and Decoding Complexity

Table 2 compares the complexities of different methods. Since VVC, Hu2020 [22], and cheng2020 [11] only run on CPU, to ensure fairness, we evaluate the running time of different methods on an 2.9GHz Intel Xeon Gold 6226R CPU. The average time at low bit rate and high bit rate over all images on Kodak dataset is used. The average model sizes at low bit rates and high bit rates are also reported.

It can be seen from Table 2 that compared to some learned methods, the proposed scheme achieves better results while providing nearly 4 times and 15 times improvement in decoded runtime at low bit rates and high bit rates, respectively (i.e., a decoding time savings of 73.35 % and 93.44 % at low bit rates and high bit rates compared with that in Cheng2020 [11]) on Kodak dataset, respectively. Our model
Learned Image Compression with Generalized Octave Convolution and Cross-Resolution Parameter Estimation

Figure 7: Example 2 in the Kodak dataset (bpp, PSNR(dB), MS-SSIM).

4.4. Comparison

The average MS-SSIM and PSNR performance over the 24 Kodak images are illustrated in Fig. 4. Guo2022 [19] achieves best results and outperforms VVC (4:4:4) in PSNR. Zhu2022 [45] is slightly worse than Guo2022 [19] and it still obtains better performance than VVC (4:4:4) in PSNR. Ho2021 [21], Ma2022 [29], and our method achieve the same performance at low bit rates. Ma2022 [29] is slightly better than Ho2021 [21] and our proposed method. The proposed scheme outperforms Lee2019 [25], Lin2020 [27], Chen2021 [9], Hu2021 [22], Li2020 [26], VVC (4:2:0) [15], BPG (4:4:4), JPEG2000, WebP, JPEG in terms of both PSNR and MS-SSIM metrics. When optimized for PSNR, VVC (4:4:4) achieves the best result. Cheng2020 [11] achieves best result in learned methods. Our proposed method almost achieves the same performance with Cheng2020 [11], and it is slightly worse than VVC (4:4:4) [15] at low bit rates. When optimized for MS-SSIM, Gu2020 [19] optimized for MS-SSIM achieves the best performance in all methods. Cheng2020 size is slightly bigger than Cheng2020 [11] and Akbari2021 [3], and also are smaller than other learned methods. Compared to Chen2021 [9], the proposed scheme provides nearly 50 times and 400 times improvement in encoding and decoded runtime at low bit rates and high bit rates on Kodak dataset, respectively. Compared to Akbari2021 [3], the proposed scheme provides nearly 10 times and 25 times improvement in encoding and decoded runtime at low bit rates and high bit rates on Kodak dataset, respectively. Compared to VVC, our encoding is about 58 times and 105 times faster on Kodak dataset, but the decoding is much slower. Our scheme has greater advantages in encoding time and decoding time in Tecnick dataset compared with other learned methods. Therefore the proposed scheme achieves the new state of the art in learned image coding when considering both the time complexity and space complexity.
Figure 8: Example 3 in the Tecnick dataset (bpp, PSNR(dB), MS-SSIM).

[11] is slightly worse than Gu2020 [19]. The proposed MS-SSIM method and Ho2021 [21] almost obtain the same performance with Cheng2020 [11] at high bit rates and achieve slightly worse performance than [11] at low bit rates. We calculate the BD-rate (%) of different methods including Chen2021 [9], Cheng2020 [11], Lee2019 [25], Hu2021 [23], Akbari2021 [3] and GLLMM [18]. The best traditional codec VVC (4:4:4) [15] is considered as anchor. Since some methods are not available, we do not compare them. The result is illustrated in Table 3. From Table 3, we can find that the GLLMM achieves an average of 4.7 % bitrate saving over VVC (4:4:4) on Kodak dataset. Our method is very close to VVC (4:4:4) and has 5.6 % bitrate increase.

We also evaluate the performance of the proposed method on the Tecnick dataset compared to traditional methods and some latest learned methods in Fig. 5. Some methods are not included because we do not have their results. When optimized for PSNR, Zhu2022 [45] achieves best results at
Learned Image Compression with Generalized Octave Convolution and Cross-Resolution Parameter Estimation

Table 3
The BD-rate (%) of different methods. The negative numbers represent bitrate saving and positive numbers indicate bitrate increase.

| Methods          | Chen2021 [9] | Cheng2020 [11] | Lee2019 [25] | Hu2021 [23] | Akbari2021 [3] | GLLMM [18] | Ours  |
|------------------|--------------|----------------|--------------|-------------|---------------|------------|-------|
| BD-rate          | 8.6          | 4.8            | 17.0         | 11.1        | 9.4           | -3.13      | 5.6   |

Table 4
The proportion of Low bit rates and the high bit rates.

| Name  | Total BPP | Low BPP | High Bpp | Ratio   |
|-------|-----------|---------|----------|---------|
| scheme 1 | 0.1790    | 0.0169  | 0.1620   | 1:9.5858|
| scheme 2 | 0.1785    | 0.0908  | 0.0877   | 1:0.9658|
| scheme 1 | 0.3440    | 0.0360  | 0.3080   | 1:8.556  |
| scheme 2 | 0.3390    | 0.1413  | 0.1977   | 1:1.399  |
| scheme 1 | 0.6850    | 0.0991  | 0.5859   | 1:5.914  |
| scheme 2 | 0.6817    | 0.2073  | 0.4744   | 1:2.288  |

Figure 9: The R-D curves of several variants of the proposed modules on the Kodak dataset. GoConv represents the modulated generalized octave convolution module. CPRE indicates the cross-resolution parameter estimation module. Loss denotes the information-fidelity loss.

To demonstrate the proposed method can generate more visually pleasant results, three visual examples from the Kodak and Tecnick dataset are illustrated in Fig. 6, Fig. 7, and Fig. 8, in which our proposed results are compared with JPEG, JPEG2000 and webp, BPG (4:4:4), JPEG2000, WebP, JPEG in term of both PSNR at all rate points. When optimized for MS-SSIM, our method achieves the best results than traditional methods and learned methods across a wide bit rates.

To demonstrate the proposed method can generate more visually pleasant results, three visual examples from the Kodak and Tecnick dataset are illustrated in Fig. 6, Fig. 7, and Fig. 8, in which our proposed results are compared with JPEG, JPEG2000 and webp, BPG (4:4:4), JPEG2000, WebP, JPEG in term of both PSNR at all rate points. When optimized for MS-SSIM, our method achieves the best results than traditional methods and learned methods across a wide bit rates.

4.5. The Different Structure Of The Two Different Hyper Layers

The first ablation study is conducted to prove the effectiveness of different components in our proposed framework in Fig. 9. In this experimental setting, we investigate the effectiveness of the proposed modules including the additional hyper layer, the cross-resolution parameter estimation (CRPE) modules, and the information-fidelity loss. The original two-layer hyper-prior model with the modulated generalized octave convolution module is considered as the baseline method. Based on the baseline, we combined different modules to the baseline to investigate the performance improvement level of different modules. To compare as fair as possible, the parameter $\lambda$ is chosen in the set $\{0.003, 0.008, 0.03\}$. The number of filters $N$ is set to 128 for all bit rates. The PSNR metric is optimized as the distortion function.

As seen in Fig. 9, the baseline model achieves poor performance. We first add an additional hyper layer to the baseline. Compared to the baseline, the Three Layers (GoConv) slightly improves the R-D performance. Then we add three CRPE modules to the Three Layers (GoConv+CPRE) scheme. We can find the CRPE modules significantly improves the compression performance. Last, we add the information-fidelity loss to the Three Layers method with CRPE modules (GoConv+CPRE+Loss) scheme. The information-fidelity loss slightly improves image compression performance. However, it significantly increases the LR portion in of the whole bit stream.

We compare the bit rate allocation of these two schemes. The comparison results are shown in Table 4. The Three Layers (GoConv+CPRE) and the Three Layers (GoConv+CPRE+Loss) are denoted as scheme 1 and scheme 2, respectively. We find if the information-fidelity loss function is adopted in the proposed framework, the ratio of LR bit rates improves a lot.

We also explore the impact of the different hyper structures on the image compression performance of the proposed
Learned Image Compression with Generalized Octave Convolution and Cross-Resolution Parameter Estimation

Table 5
The different hyper structures of different schemes. HE denotes Hyper Encoder, and HD represent Hyper Decoder.

| Scheme 1 | First Layer | Second Layer | Third Layer | Scheme 2 | First Layer | Second Layer | Third Layer |
|----------|-------------|--------------|-------------|----------|-------------|--------------|-------------|
| HE \(h_1\) | 3 × 3, N, s1 | 3 × 3, N, s2 \(\downarrow\) | 3 × 3, N, s2 \(\downarrow\) | HE \(h_1\) | 3 × 3, N, s1 | 3 × 3, N, s2 \(\downarrow\) | 3 × 3, N, s1 |
| HE \(f_1\) | 3 × 3, N, s2 | 3 × 3, N, s1 | 3 × 3, N, s1 | HE \(f_1\) | 3 × 3, N, s1 | 3 × 3, N, s2 \(\downarrow\) | 3 × 3, N, s1 |
| HD \(f_2\) | 3 × 3, N, s1 | 3 × 3, N, s1 | 3 × 3, N, s1 | HD \(f_2\) | 3 × 3, N, s1 | 3 × 3, N, s2 \(\uparrow\) | 3 × 3, N, s1 |
| HD \(h_2\) | 3 × 3, N, s1 | 3 × 3, N, s2 \(\uparrow\) | 3 × 3, N, s2 \(\uparrow\) | HD \(h_2\) | 3 × 3, N, s1 | 3 × 3, N, s2 \(\uparrow\) | 3 × 3, N, s1 |

Table 6
The different structure of two different hyper layers on Kodak dataset.

| Name  | \(N\) | \(\lambda\) | Bpp | PSNR(dB) | MS-SSIM |
|-------|-------|-------------|-----|----------|---------|
| Scheme 1 | 128 | 0.002 | 0.1343 | 28.36 | 0.9224 |
| Scheme 2 | 128 | 0.002 | 0.1442 | 28.12 | 0.9204 |
| Scheme 1 | 256 | 0.002 | 0.1311 | 28.22 | 0.9224 |
| Scheme 2 | 256 | 0.002 | 0.1977 | 28.14 | 0.9212 |

method. Scheme 1 is the proposed method that the hyper encoder \(h_e\) have two downsampling layers. Scheme 2 is that the hyper encoder \(h_e\) and hyper encoder \(h_i\) both have one down-sampling layer. The detailed structures are shown in Table 5. \(3 \times 3, N, s2(s1), \downarrow(\uparrow)\) indicates the size of the convolution kernel, the number filters, convolution step size, down-sampling (up-sampling) operation, separately. We train four sets of the models.

The detailed parameters and result are shown in Table 5 and Table 6. First, scheme 1 outperforms scheme 2 in terms of both PSNR and MS-SSIM. The hyper encoder \(h_e\) just have the one down-sampling layer, the latent representations of the \(h_e\) will occupy a larger bit rate ratio than \(h_i\) which have two down-sampling layers. The hyper layer is just an auxiliary layer, which can learn side information to capture spatial dependencies among the elements of the latent representations in Encoder \(g_e\). Therefore, there is no need to spend too much bits to encode the latent representations of the \(h_e\). Second, at low bit rates, the image compression performance will not improve significantly by increasing the number of filters. It only takes a few channels to learn the relevant features of the image. However, as the bit rate increases, if the number of channels is not increased, the image compression performance will decrease significantly. To achieve better R-D performance, most approaches used different channel numbers to train models to obtain different bit rates.

5. Conclusion
As mentioned before, the learned methods based on context-adaptive entropy models achieve state-of-the-art performance among all the learned methods, in which the hyper-prior and autoregressive models are jointly optimized to significantly remove the spatial dependencies of the latent representations. In this paper, we presented a multi-layer hyper-prior and cross-resolution parameter estimation image compression method based on generalized octave convolution. We remove the context models from the image compression to improve the parallelism of decoding, which will reduce the R-D performance and lead to a strong spatial correlation in latent representations. We improve the performance of the compression network from three aspects. First, the generalized octave convolution was introduced to factorize the latent representations into high and low resolution. Second, an additional hyper-layer is introduced to further remove the spatial redundancy of the latent representation. Third, to allow the undecoded components to utilize the previously decoded information as much as possible, the cross-resolution parameter estimator (CRPE) modules are introduced to improve the R-D curve performance.

Experiments show that the proposed scheme without context models can outperform VVC (4:2:0) and some recent learned methods in terms of both PSNR and MS-SSIM metrics across a wide bit rates. However, the proposed scheme can still have some room to improve. For example, we can exploit more efficient probability models to accurately establish the probability distribution of potential representations. The complexity of the proposed method can be further optimized by different approaches such as model compression and optimization.

References
[1] Agustsson, E., Mentzer, F., Tschannen, M., Cavigelli, L., Timofte, R., Benini, L., Gool., L.V., 2017. Soft-to-hard vector quantization for end-to-end learning compressible representations, in: In Advances in Neural Information Processing Systems, pp. 1141–1151.
[2] Agustsson, E., Tschannen, M., Mentzer, F., Timofte, R., Van Gool, L., 2019. Generative adversarial networks for extreme learned image compression, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 221–231. doi:10.1109/ICCV.2019.00031.
[3] Akbari, M., Liang, J., Han, J., Tu, C., 2021. Learned bi-resolution image coding using generalized octave convolutions, in: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 6592–6599.
[4] Bai, L., Liu, Q., Li, C., Ye, Z., Hui, M., Jia, X., 2022. Remote sensing image scene classification using multi-scale feature fusion covariance network with octave convolution. IEEE Transactions on Geoscience and Remote Sensing, 1–10. doi:10.1109/TGRS.2022.3168492.
[5] Ballé, J., Laparra, V., Simoncelli, E.P., 2016. End-to-end optimization of nonlinear transform codes for perceptual quality, in: 2016 Picture Coding Symposium (PCS), pp. 1–5.
[6] Ballé, J., Laparra, V., Simoncelli, E.P., 2017. End-to-end optimized image compression, in: International Conference on Learning Representations.
[7] Ballé, J., Minnen, D., Singh, S., Hwang, S., Johnston, N., 2018.
