Selective compression learning of latent representations for variable-rate image compression

Anonymous Author(s)
Affiliation
Address
email

Abstract

Recently, many neural network-based image compression methods have shown promising results superior to the existing tool-based conventional codecs. However, most of them are often trained as separate models for different target bit rates, thus increasing the model complexity. Therefore, several studies have been conducted for learned compression that supports variable rates with single models, but they require additional network modules, layers, or inputs that often lead to complexity overhead, or do not provide sufficient coding efficiency. In this paper, we *firstly* propose a selective compression method that partially encodes the latent representations in a fully generalized manner for deep learning-based variable-rate image compression. The proposed method adaptively determines essential representation elements for compression of different target quality levels. For this, we first generate a 3D importance map as the nature of input content to represent the underlying importance of the representation elements. The 3D importance map is then adjusted for different target quality levels using importance adjustment curves. The adjusted 3D importance map is finally converted into a 3D binary mask to determine the essential representation elements for compression. The proposed method can be easily integrated with the existing compression models with a negligible amount of overhead increase. Our method can also enable continuously variable-rate compression via simple interpolation of the importance adjustment curves among different quality levels. The extensive experimental results show that the proposed method can achieve comparable compression efficiency as those of the separately trained reference compression models and can reduce decoding time owing to the selective compression.

1 Introduction

Recently, neural network (NN)-based image compression methods [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13] have been actively studied and shown superior performance in terms of PSNR BD-rate to those of the conventional tool-based compression methods, such as BPG [14] and JPEG2000 [15]. A few recent methods [11, 12] achieved comparable results with respect to the state-of-the-arts codec, called H.266 Intra-coding [16]. However, since most of the previous deep learning-based models are trained separately for different target compression levels, several models with a large number of parameters are required to support various compression levels. To address this issue, recently several methods [1, 17, 18, 19, 20, 21, 22], which use conditional transform or adaptive quantization, have been proposed. However, most of them require additional network modules, layers, or inputs that may cause complexity overhead. In this paper, a novel 'selective compression of representations' (SCR) method is presented, which performs entropy coding only for the partially selected latent

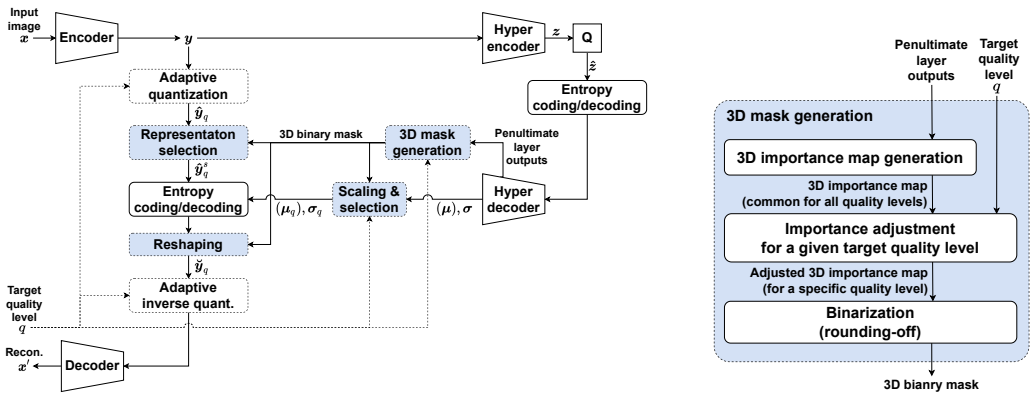
Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

representations. The selection of representations is determined via a 3D binary mask generation process in a target quality-adaptive manner. In the 3D binary mask generation of our SCR method (see Fig. 1b), (i) a 3D importance map of the same size, *independent of target quality levels*, is generated for the multi-channel feature maps (3D representations); (ii) the 3D importance map is adjusted via the channel-wise importance adjustment curves for a given target quality level; and (iii) the 3D binary mask is then generated by taking the round-off the adjusted 3D importance map. Note that the *target-quality-independent* 3D importance map becomes *target-quality-dependent* after the channel-wise importance adjustment. We incorporate our method with an adaptive quantization scheme similar to Cui *et al.* [19] and Chen *et al.* [23], into several existing deep learning-based reference compression models [6, 7], where we jointly optimize the whole elements together with our selective compression of latent representations and adaptive quantization in an end-to-end manner. In the architectural aspect, our SCR method minimizes the overhead by utilizing only a single 1×1 convolutional layer to generate the 3D importance map and importance adjustment curves for a finite number of target quality levels. In addition, the SCR method also supports continuously variable-rate compression through a simple non-linear interpolation of the importance adjustment curves between two discrete target quality levels. Furthermore, the SCR method reduces the decoding time compared to both the reference compression models [6, 7] and the very lightweight adaptive quantization-based variable-rate method similar to Cui *et al.* [19] by skipping the entropy decoding process for a substantial amount of unselected representations. Impressively, the coding efficiency of the proposed SCR method is better or comparable to those of the separately trained reference compression models [6, 7], for different target quality levels, and is superior to that of the adaptive quantization-based method. The contributions of this study are summarized as follows:

- To our best knowledge, the proposed SCR method is the first NN-based variable-rate image compression method that selectively compresses the representations in a fully generalized way and a target quality-adaptive manner. It provides compression efficiency comparable to those of the separately trained reference compression models.
- The proposed SCR method can be applied to various existing image compression models without modifying their architectures, thus allowing for a high applicability. We incorporate very lightweight modules for SCR, including only a single 1×1 convolutional layer and a small number of importance-adjustment curves, into the existing compression models. Our SCR method even reduces a decoding time compared to those of the existing lightweight variable-rate model and reference compression models owing to the selective compression.
- The proposed SCR method enables continuously variable-rate compression by a simple interpolation of the importance-adjustment curves between discrete quality levels in which the selective compression is trained.

2 Related work

Several studies [1, 17, 18, 19, 23, 20, 21, 24, 22] have been conducted for enabling a single NN-based image compression model to support variable-rate compression. The first learning-based variable-rate image compression model [1] progressively performs image compression in a low-to-high quality manner by accumulating additional binary representations as the number of compression iterations increases. For the entropy model-based approaches, Choi *et al.* [17] proposed a conditional convolution that adaptively operates according to different target quality levels. Scale and shift factors are derived from a one-hot vector representing a quality level, and are then applied to each output of convolutional layers, which may cause to increase complexity overhead. Cai *et al.* [18] stacks representations to form a multiscale structure, and then determine point-wise scales, representing the importance levels of representations to be compressed. This method also uses additional modules named a multi-scale decomposition transform layer (MSD) and an inverse multi-scale decomposition transform layer (IMSD), both of which consist of multiple convolutional layers. Cui *et al.* [19] and Chen *et al.* [23] utilize two types of vectors for the adaptive quantization and inverse quantization, respectively. These methods support variable-rate compression with negligible overhead on top of the



(a) (b)

Figure 1: (a) The overall architecture of the proposed SCR method. In this figure, the SCR method is incoporated into the Hyperprior [6] model. The elements for the variable-rate compression are represented as the dotted boxes, and especially those for the selective compression are highlighted in light blue. (b) The 3D binary mask generation process of the SCR method.

existing compression models ([6, 7] for Cui *et al.* [19] and [3, 6, 10] for Chen *et al.* [23]). However, as described in our experimental results and Ripple's work [20], the adaptive quantization alone does not provide sufficient coding efficiency compared to those of the separately trained compression models. Lu *et al.* [21] use two additional modules called an adaptive quantization layer (AQL) and an inverse adaptive quantization layer (IAQL) to obtain quantization and inverse quantization factors for each representation. Here, the AQL and IAQL layers are separately trained for each target quality level. Ripple *et al.* [20] additionally feed the information of a target quality level into each convolutional layer in the form of a "level map" where a one-hot vector of 8 quality levels is assigned for all spatial positions. The input level map can make each convolutional layer adaptively work according to a given target quality level at the cost of increased computation complexity. Lu *et al.* [24] presented an NN-based quality scalable coding scheme, named PLONQ, using nested quantization and latent ordering. However, it could not achieve comparable results to its separately trained base compression model [7]. Song *et al.* [22] supports the image compression of spatially different qualities using a spatial quality map. Although this method provides a new functionality, the additional module, named spatially-adaptive feature transform, that transforms the spatial quality map into an input for each convolutional layer may increase the overall complexity.

3 Proposed method

3.1 Overall architecture

Our SCR method can be combined with adaptive quantization such as Cui *et al.* [19] and Chen *et al.* [23], on top of several compression architectures with hyper-encoder and hyper-decoder [6] such as the models in [6, 7, 8, 9, 10, 11, 12], as shown in Fig. 1a. In this paper, we apply our SCR method for several reference compression models, Hyperprior [6], Mean-scale [7], and Context [7], to show its effectiveness with generality. In the architecture with hyper-encoder and hyper-decoder [6], input image $\boldsymbol{x}$ is transformed into a representation $\boldsymbol{y}$ using an encoder network, and the hyper encoder and decoder are used to code the distribution parameters for the quantized representation $\hat{\boldsymbol{y}}$ of $\boldsymbol{y}$ as a side information, with which $\hat{\boldsymbol{y}}$ is entropy-coded and decoded. The representation $\hat{\boldsymbol{y}}$ is then reconstructed into an image $\boldsymbol{x}'$ through a decoder network. Upon this base compression architecture, we exploit two additional elements: adaptive quantization and selective compression to enable variable-rate compression. The selection of representation elements in the encoder side is expressed as follows:

$$\hat{\boldsymbol{y}}_q^s = M(\hat{\boldsymbol{y}}_q, m(\hat{\boldsymbol{z}}, q)), \text{ with } \hat{\boldsymbol{y}}_q = AdaQ_q(\boldsymbol{y}), \tag{1}$$

where $\hat{\boldsymbol{y}}_q^s$ is a set of the selected elements of $\hat{\boldsymbol{y}}_q$ for a given target quality level q and $AdaQ_q(\cdot)$ is a target quality-adaptive quantization operator where $AdaQ_q(\boldsymbol{y}) = Round(\boldsymbol{y}/\boldsymbol{QV}_q)$ with a quantization vector $\boldsymbol{QV}_q$. $M(\cdot)$ is an element selection operator for $\hat{\boldsymbol{y}}_q$, and $m(\hat{\boldsymbol{z}}, q)$ represents a

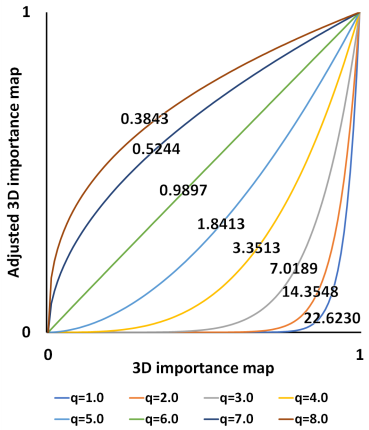


Figure 2: Some examples of importance adjustment curves in 8 target quality levels. For this experiment, the proposed SCR method was implemented on top of the Hyperprior [6] model.

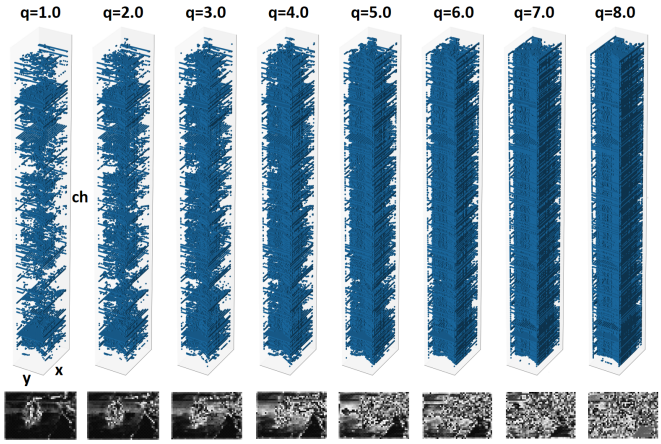


Figure 3: Top - sample masks for 8 target quality levels where the dark blue indicates the selected representation elements by 3D binary masks. Bottom - the masks averaged along the channel axis. The higher the target quality is, the more the representation elements are selected, especially in more complex regions.

generated 3D binary mask for q and quantized hyperprior $\hat{\boldsymbol{z}}$ (Sec. 3.2). The representation $\boldsymbol{y}$ is the output $En(\boldsymbol{x})$ of the encoder network $En(\cdot)$ for input image $\boldsymbol{x}$ in Fig. 1a. It should be noted that $\hat{\boldsymbol{y}}_q^s$ is entropy-coded and entropy-decoded using an entropy model based on a target quality dependent distribution P_q (Sec. 3.3). The image reconstruction $\boldsymbol{x}'_q$ in the decoder side is given as:

$$\boldsymbol{x}'_q = De(AdaIQ_q(\breve{\boldsymbol{y}}_q)), \quad \text{with } AdaIQ_q(\breve{\boldsymbol{y}}_q) = \breve{\boldsymbol{y}}_{\boldsymbol{q}} \cdot \boldsymbol{IQV}_q \text{ and } \breve{\boldsymbol{y}}_q = Re(\hat{\boldsymbol{y}}_q^s, m(\hat{\boldsymbol{z}}, q)), \tag{2}$$

where $\boldsymbol{x}'_q$ is a reconstructed image for a given target quality level q, as the output of the decoder network $De(\cdot)$, $AdaIQ_q(\cdot)$ is an adaptive inverse-quantization operator that multiplies an inverse-quantization vector $\boldsymbol{IQV}_q$ to input $\breve{\boldsymbol{y}}_q$, and $Re(\cdot)$ is a reshaping operator that converts the selected elements $\hat{\boldsymbol{y}}_q^s$ in a 1D shape into the elements of a 3D-shaped representation in place by using the 3D binary mask $m(\hat{\boldsymbol{z}}, q)$. For the unselected elements, the reshaping operator $Re(\cdot)$ places zeros in the corresponding positions. Example source codes for $M(\cdot)$ and $Re(\cdot)$ are provided in Appendix A.1.

In Eqs. 1 and 2, the vector dimensionalities of $\boldsymbol{QV}_q$ and $\boldsymbol{IQV}_q$ are the same as $C_{\boldsymbol{y}}$, the number of channels in $\boldsymbol{y}$, so that the quantization of $\boldsymbol{y}$ and the inverse quantization of $\breve{\boldsymbol{y}}_q$ are performed channel-wise by the respective elements of $\boldsymbol{QV}_q$ and $\boldsymbol{IQV}_q$, respectively, as in the previous adaptive quantization method [19].

3.2 3D binary mask generation

The 3D binary mask generation process consists of the three steps: (i) 3D importance map generation, (ii) importance adjustment, and (iii) binarization, as shown in Fig. 1b, which is given as:

$$m(\hat{\boldsymbol{z}}, q) = B(im(\hat{\boldsymbol{z}})^{\boldsymbol{\gamma}_q}) \tag{3}$$

where $im(\hat{\boldsymbol{z}})$ is a 3D importance map that is generated via the hyper-decoder for the hyperprior $\hat{\boldsymbol{z}}$ as input, $\boldsymbol{\gamma}_q = \left[\gamma_q^1, \gamma_q^2, ..., \gamma_q^N\right]$ is a parameter vector of dimensionality $N(= C_{\boldsymbol{y}})$ where the parameters are learned to determine the channel-wise importance adjustment curves for a given target quality q, and $B(\cdot)$ is a binarization operator with the rounding-off.

3D importance map generation. The 3D importance map $im(\hat{\boldsymbol{z}})$, which has values in the range between 0 and 1, represents the underlying importance of each element in $\boldsymbol{y}$. Without utilizing a dedicated complex network that generates $im(\hat{\boldsymbol{z}})$, we feed the outputs of the penultimate convolutional layer (after the activation) in the hyper decoder into a single 1×1 convolutional layer in the mask generation module, followed by a clipping function to obtain the importance values between 0 and 1. Note that the 3D importance map is generated, not dependently of target quality levels but dependently of input images, thus it represents the nature of $\boldsymbol{y}$ in perspective of element-wise importance.

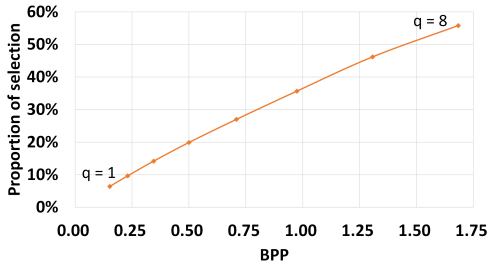


Figure 4: Average proportions of selected representation elements versus average BPP. (test set: the Kodak image set [25], base model: Hyperprior [6])

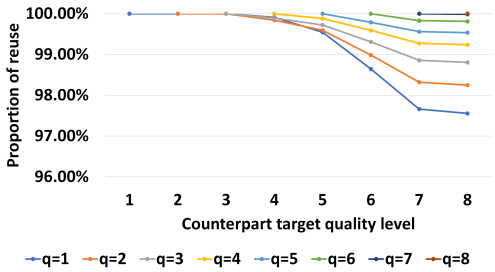


Figure 5: Average proportions of reused representation elements from low to high quality levels. (test set: the Kodak image set [25], base model: Hyperprior [6])

Importance adjustment. The actual importance of each representation element may vary according to various target quality levels. For example, some representation elements corresponding to the texture of high complexity in the images may not be necessarily required in low-quality compression. Thus, it is natural to adjust the 3D importance map, which is commonly used for all quality levels, according to a specific target quailty level. For this, we devise a scheme of adjusting the 3D importance map $im(\hat{\boldsymbol{z}})$ using importance adjustment curves for various target quality levels. The importance adjustment curves change the element values of $im(\hat{\boldsymbol{z}})$ channel-wise where their curvatures are learned as a parameter vector $\boldsymbol{\gamma}_q$ for $1 < q < N_Q$ where N_Q is a total number of target quality levels. Note that the target quality increases as q increases. Fig. 2 shows some examples of importance adjustment curves. In Fig. 2, the horizontal and vertical axes represent the input $im(\hat{\boldsymbol{z}})$ value to be adjusted and its adjusted result, respectively, and the numbers labeled on the importance adjustment curves indicate the average values of trained $\boldsymbol{\gamma}_q$ vectors for N_Q target quality levels. It is noted in Fig. 2 that the importance adjustment curves for $q > 6$ tend to amplify the elements of input $im(\hat{\boldsymbol{z}})$ while attenuating them for $q < 6$ in an average sense. For $q = 6$, there is little variation with an average $\bar{\boldsymbol{\gamma}}_q = 0.9897$ before and after the importance adjustment. Consequently, $im(\hat{\boldsymbol{z}})$ is more strongly amplified in an overall sense for higher target quality levels. Whereas, for the lower target quality levels, $im(\hat{\boldsymbol{z}})$ is largely attenuated in general, so only a small number of $im(\hat{\boldsymbol{z}})$ elements whose values are close to 1 can maintain their importance. The total number of $\boldsymbol{\gamma}_q$ vectors is N_Q, thus a total of $N_Q \times C_{\boldsymbol{y}}$ parameters are learned for all $\boldsymbol{\gamma}_q$ vectors. In our implementation, N_Q is set to 8 and $C_{\boldsymbol{y}}$ is set to that of the original reference model.

Binarization. The 3D binary mask is finally determined by the rounding operator, denoted as $B(\cdot)$. Here, "1" values in the output 3D binary mask indicate that the corresponding elements in $\boldsymbol{y}$, at the same coordinates, are selected for compression. Fig. 3 shows examples of the generated masks for different target quality levels from $q = 1.0$ to $q = 8.0$ when our SCR method was implemented on top of the Hyperprior [6] model and Kodim12 image of the Kodak image set [25] is used as an input sample, in which the components marked in dark blue indicate "1" values. When q is set to 1.0, the lowest quality level in our SCR method, only 3.22% of the total elements are selected, and the selection ratio gradually increases as the q value increases. For $q = 8.0$, 43.39% of the representation elements are selected. In addition, as shown in the averaged masks along the channel axis, the proposed method uses more representations in the high-complexity region. Over the whole Kodak image set [25], the average proportions of selected elements for the target quality levels from 1.0 to 8.0 were 6.41%, 9.66%, 14.17%, 19.90%, 27.00%, 35.68%, 46.20%, and 55.81%, respectively, where they are almost linearly proportional to the average bpp values, as shown in Fig. 4.

Fig. 5 shows how many representations in a low target quality level are commonly used (or selected) for higher target quality levels. For example, the orange line indicates that 100%, 99.8%, 99.6%, 99.0%, 98.3% and 98.2% of the representation elements, which are selected for a target quality level $q = 2.0$, are also reused for higher target quality levels from $q = 3.0$ to 8.0, respectively. It should be noted in Fig. 5 that the case with $q = 8.0$ tends to highly reuse 97.6% of the selected representation elements for the case with $q = 1.0$. This implies that the proposed SCR method does not select the representation elements separately for different target quality levels, but actively takes a large portion of representation elements as common components for various target quality levels.

3.3 Training

We train the proposed SCR model in an end-to-end manner using the total loss formulated as follows:

$$\mathcal{L} = \sum_q R_q + \lambda_q * D_q, \quad \text{with } R_q = H_q(\tilde{\boldsymbol{y}}_q^s \mid \tilde{\boldsymbol{z}}) + H(\tilde{\boldsymbol{z}}), \tag{4}$$

where R_q and D_q represent rate and distortion terms, respectively, for the target quality level q, and $\lambda_q = 0.2 \cdot 2^{q-8}$ is a parameter for adjusting the balance between the rate and distortion. D_q can be eighter the mean squared error (MSE) or MS-SSIM between the input image $\boldsymbol{x}$ and the reconstructed image $\boldsymbol{x}'_q$. For the MS-SSIM-based optimization, we use the distortion term $D_q = 3000(1 - \textit{MS-SSIM}(\boldsymbol{x}, \boldsymbol{x}'_q))$. $H(\cdot)$ is a calculated cross-entropy for the quantized representations of $\boldsymbol{y}$ and $\boldsymbol{z}$. In the case of $\boldsymbol{y}$, because the quantization and mask generation processes are different for each target quality level q, we use a cross entropy $H_q(\cdot)$ for a target quality level q as follows:

$$H_q(\tilde{\boldsymbol{y}}_q^s \mid \tilde{\boldsymbol{z}}) = \frac{1}{N_{\boldsymbol{x}}} \sum_{i=1}^{N_q^s} -\log_2 P_q(\tilde{y}_{q,i}^s | \hat{\boldsymbol{z}}), \tag{5}$$
$$\text{with } \tilde{\boldsymbol{y}}_q^s = M(\boldsymbol{y}/\boldsymbol{QV}_q + U(-0.5, 0.5), \tilde{m}(\hat{\boldsymbol{z}}, q)),$$

where $N_{\boldsymbol{x}}$ is the number of pixels in a input image $\boldsymbol{x}$, N_q^s is a total number of the selected elements $\{\tilde{y}_{q,i}^s\}_{i=1}^{N_q^s}$ of $\tilde{\boldsymbol{y}}_q^s$. The cross entropy, $H_q(\tilde{\boldsymbol{y}}_q^s \mid \tilde{\boldsymbol{z}})$, of the selected representation elements is calculated based on an approximate probability mass function (PMF) $P_q(\cdot)$ to deal with the distributions of $\tilde{\boldsymbol{y}}_q^s$ that vary for different target quality levels. Specifically, we determine the estimated distribution parameters $\boldsymbol{\mu}_q$ and $\boldsymbol{\sigma}_q$ of $P_q(\cdot)$ as $M(\boldsymbol{\mu}/\boldsymbol{QV}_q, m(\hat{\boldsymbol{z}}, q))$ and $M(\boldsymbol{\sigma}/\boldsymbol{QV}_q, m(\hat{\boldsymbol{z}}, q))$, respectively. Here, the $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ values are obtained from the base compression models. For the Context [7] base model, the position-wise parameters $\boldsymbol{\mu}_q^{(k,l)}$ and $\boldsymbol{\sigma}_q^{(k,l)}$ are obtained for each spatial coordinate (k, l) through $M(\boldsymbol{\mu}^{(k,l)}/\boldsymbol{QV}_q, m(\hat{\boldsymbol{z}}, q)^{(k,l)})$ and $M(\boldsymbol{\sigma}^{(k,l)}/\boldsymbol{QV}_q, m(\hat{\boldsymbol{z}}, q)^{(k,l)})$, respectively. Note that we disregard $\boldsymbol{\mu}_q$ when a zero-mean Gaussian-based model is used for $P_q(\cdot)$. As in the previous entropy minimization-based compression models [6, 7], we adopt a Gaussian distribution model convolved with a uniform distribution as an approximate PMF $P_q(\cdot)$, and use the representation with additive uniform noise $U(-0.5, 0.5)$, denoted as $\tilde{\boldsymbol{y}}_q^s$, for training, rather than the rounded representation $\hat{\boldsymbol{y}}_q^s$ for inference. To handle the instability in the training phase due to learning the binary representations of the mask, we use a stochastically generated mask $\tilde{m}(\cdot)$ rather than $m(\cdot)$ used in the test phase. While the adjusted 3D importance map is simply rounded-off for $m(\cdot)$, $\tilde{m}(\cdot)$ is constructed with randomly sampled binary representations, similarly to Raiko *et al.* [26]'s approach, by regarding each element value of the adjusted 3D importance map $im(\hat{\boldsymbol{z}})^{\gamma_q}$ as the probability that the corresponding component of the output mask is "1", which is given as follows:

$$\tilde{m}(\hat{\boldsymbol{z}}, q) = B(im(\hat{\boldsymbol{z}})^{\gamma_q} + U(-0.5, 0.5)) \tag{6}$$

Note that discontinuity caused by the rounding-off operator $B(\cdot)$ is handled by bypassing the gradients backward. In the actual implementation, the training can be performed without using $M(\cdot)$ and $Re(\cdot)$, because we can exclude the unselected representations using Eq. 7 to calculate R_q, and can obtain $\breve{\boldsymbol{y}}_q$ via $AdaQ_q(\boldsymbol{y}) \cdot \tilde{m}(\hat{\boldsymbol{z}}, q)$ to compute D_q. Other training details are described in Appendix A.2

$$H_q(\tilde{\boldsymbol{y}}_q^s \mid \tilde{\boldsymbol{z}}) = \frac{1}{N_{\boldsymbol{x}}} \sum_i -\log_2 P_q(\tilde{y}_{q,i}) \cdot \tilde{m}(\hat{\boldsymbol{z}}, q)_i, \quad \text{with } \tilde{\boldsymbol{y}}_q = \boldsymbol{y}/\boldsymbol{QV}_q + U(-0.5, 0.5) \tag{7}$$

3.4 Continuously variable-rate compression

To support the continuously variable-rate compression during the test, we determine $\boldsymbol{\gamma}_q$ by interpolation when q is a value between two discrete target quality levels as follows:

$$\boldsymbol{\gamma}_q = \begin{cases} \boldsymbol{\gamma}_q, & \text{if } q \in \{1, 2, ..., N_Q\} \\ \boldsymbol{\gamma}_{\lfloor q \rfloor}^{1-(q-\lfloor q \rfloor)} \cdot \boldsymbol{\gamma}_{\lceil q \rceil}^{q-\lfloor q \rfloor}, & \text{otherwise} \end{cases} \tag{8}$$

For example, when q is 3.8, $\boldsymbol{\gamma}_{3.8}$ is determined by the element-wise multiplication of $\boldsymbol{\gamma}_{3.0}^{0.2}$ and $\boldsymbol{\gamma}_{4.0}^{0.8}$. Note that the $\boldsymbol{QV}_q$ and $\boldsymbol{IQV}_q$ vectors are also interpolated in the same manner as above, and this non-linear interpolation is inspired by G-VAE [19]. The experimental results of the continuously variable-rate are provided in Sec. 4.1.

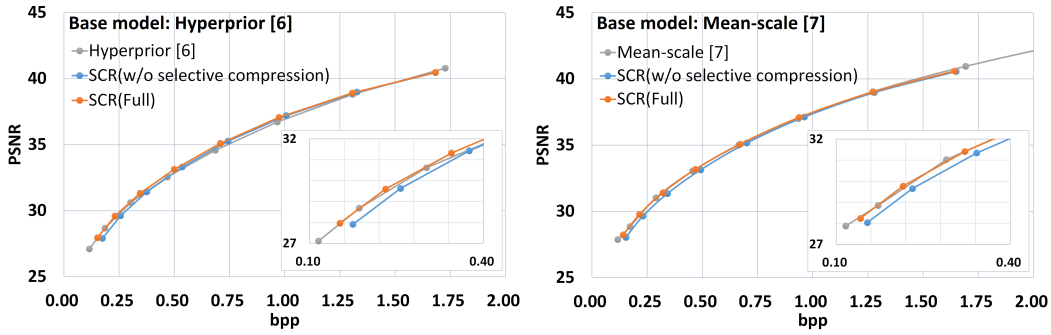


Figure 6: Rate-distortion curves in discrete target quality levels for the reference compression model, the SCR variants without the selective compression, and SCR full model. Here, MSE-optimized models are used and the results for the MS-SSIM-optimized models are provided in Appendix A.3.

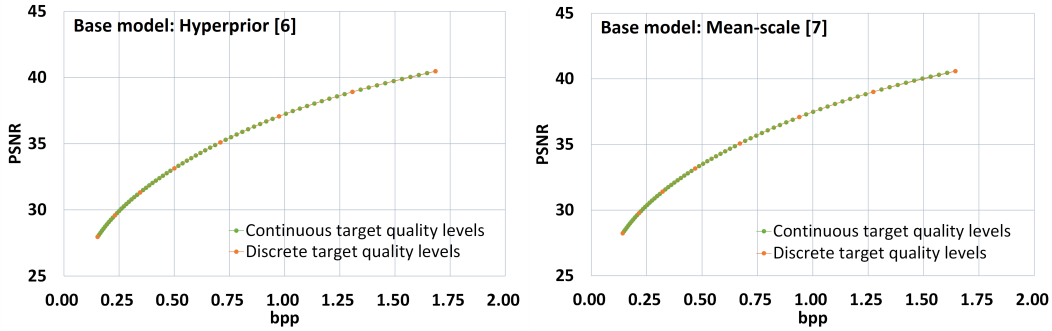


Figure 7: Rate-distortion curves of our SCR model in continuous target quality levels. The orange dots indicate the results for discrete target quality levels while the green dots represent the results using $\boldsymbol{\gamma}_q$ vector interpolation.

4 Experiments

4.1 Experiments for various target quality levels

Experiments for discrete target quality levels. To show the effectiveness of our SCR method with selective compression and adaptive quantization, we integrate it into the following three reference compression models: Hyperprior [6], Mean-scale [7] and Context [7] which are denoted as SCR_{Hyp}, SCR_{MS} and SCR_{Cxt}, respectively. These three extended models with our SCR method are compared with their original models in terms of PSNR(MS-SSIM) BD-rate. The experimental results of ablation study are also provided to show the effectiveness of our selective compression where the SCR_{Hyp}, SCR_{MS} and SCR_{Cxt} are compared with and without the selective compression modules. Note that the SCR method without the selective compression only uses the adaptive quantization for variable-rate compression. It should be noted that most of the previous variable-rate methods [1, 17, 18, 20, 21, 22] use their proprietary architectures, rather than the widely used models [6, 13], thus making it difficult the direct comparison with the SCR_{Hyp}, SCR_{MS} and SCR_{Cxt} in a same condition. For comparisons, we train each of the SCR_{Hyp}, SCR_{MS} and SCR_{Cxt} as two different versions, an MSE(PSNR)-optimized model SCR_X^{psnr} and an MS-SSIM-optimized model which are denoted as SCR_X^{psnr} and SCR_X^{ssim}, respectively. We measure average bpp-PSNR and bpp-MS-SSIM values over the Kodak image set [25]. Fig. 6 shows the comparison results for SCR_{Hyp}^{psnr} and SCR_{MS}^{psnr} . The results for SCR_{Cxt}^{psnr} are shown in Appendix A.3. Compared with the original Hyperprior [6] base model and Mean-scale [7] base model, the SCR_{Hyp}^{psnr} and SCR_{MS}^{psnr} yielded a BD-rate gain of -3.21% and a BD-rate loss of 1.30%, respectively. For the cases of SCR_{Hyp}^{psnr} and SCR_{MS}^{psnr} without the selective compression, the coding efficiency is reduced by 4.30% and 5.03% in BD-rate, respectively. This clearly demonstrates that the selective compression in the SCR method is effective in terms of the coding efficiency.

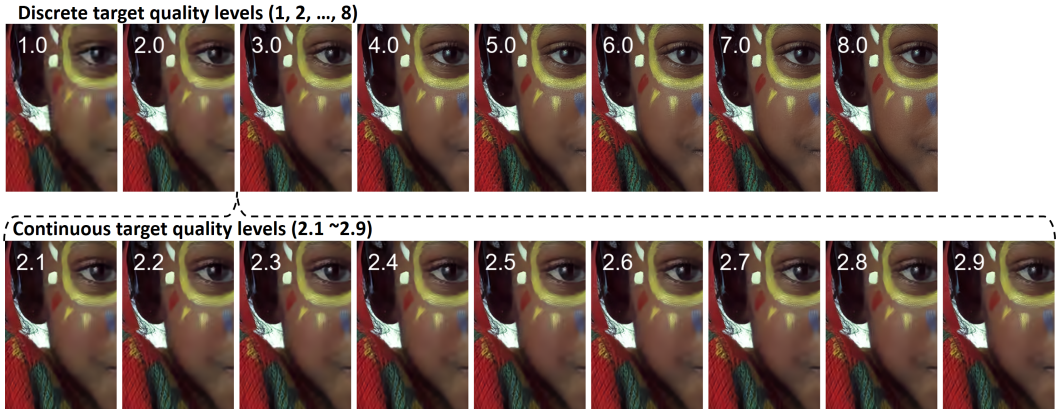


Figure 8: The qualitative results of the proposed SCR method (best viewed in digital format). For best viewing, we cropped the reconstruction results. The number embedded in each sample represents the target quality level q. The top row represents the reconstruction results of the discrete quality levels. The bottom row shows the reconstruction results between $q = 2.0$ and $q = 3.0$ with the interpolation method in Sec. 3.4.

For the Context [7] reference model, the SCR^{psnr}_{Cxt} model yielded a BD-rate loss of 1.23% and a BD-rate gain of -1.18%, respectively, compared with its reference compression model and its variant without the selective compression as shown in Appendix A.3. The relatively smaller contribution of selective compression may come from two reasons: (i) In the autoregressive Context [7] model, the reconstructed representation elements are utilized to predict the distribution parameters of the adjacent elements. Thus, in SCR^{psnr}_{Cxt}, the unselected representations filled with 0 values may deteriorate the prediction accuracy; (ii) Unlike for the other two reference compression models, Hyper [6] and Mean-scale [7], the SCR^{psnr}_{Cxt} without the selective compression already achieved a fairly satisfactory coding efficiency with respect to the original Context [7] model, so there seems to be not much room for performance improvement. Although there exist somewhat deviations in coding efficiency gain for several reference compression models, it is worthwhile to note that our SCR models achieve comparable results to all the reference models separately trained for various target quality levels.

For the MS-SSIM-optimized models, as shown in Appendix A.3, we obtained similar results as those of MSE-optimized models. The SCR^{ssim}_{Hyp}, SCR^{ssim}_{MS} and SCR^{ssim}_{Cxt} obtained a BD-rate gain of -0.91%, a BD-rate loss of 0.96%, and a BD-rate gain of -2.00%, respectively, compared with their original Hyperprior [6], Mean-scale [7], and Context [7] models. The BD-rate gains of the SCR full models, SCR^{ssim}_{Hyp}, SCR^{ssim}_{MS}, and SCR^{ssim}_{Cxt}, were -4.03%, -3.00%, and -1.22%, respectively, compared with their respective SCR variants without the selective compression. The rate-distortion curves for MS-SSIM-optimized models are provided in Appendix A.3.

Experiments for fine-grain target quality levels. To verify the continuously variable-rate compression of the proposed SCR method, we measured the rate-distortion performance by changing q from 1.0 to 8.0 with an increment of 0.1. As shown in Fig. 7, the proposed SCR method stably supports the continuously variable-rate compression without degrading coding efficiency. Fig. 8 shows the reconstructed samples of the proposed SCR method where the quality gradually improves as q increases. More reconstruction samples are provided in Appendix A.4.

4.2 Decoding time

We compare the decoding times for the original reference compression models (Hyperprior [6] and Mean-scale [7]) and their SCR extensions (SCR^{psnr}_{Hyp}, SCR^{psnr}_{MS}) for variable bitrate image compression. To show the effect of the selective compression in our SCR method on the decoding time, the decoding times of the above models, SCR^{psnr}_{Hyp} and SCR^{psnr}_{MS}, are compared for the cases with and without the selective compression in the same target quality levels. For the comparison of the two reference compression models, Hyper [6] and Mean-scale [7], their decoding times are measured for a high quality compression model that corresponds to our variable-rate configuration with $q = 8.0$. Note that, since each of the two reference compression models has different architectures for different

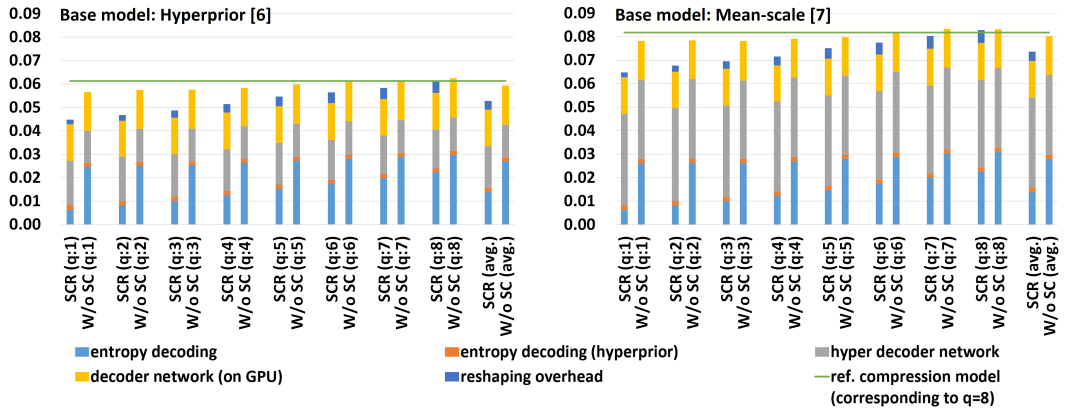


Figure 9: Decoding times of the proposed SCR models, the SCR variants without the selective compression, denoted as "W/o SC", and the reference compression models. We measured the average decoding time of each model over the Kodak image dataset [25] with 2×Intel Xeon Gold 5122 CPUs and 1×RTX Titan GPU.

target quality levels, the model corresponding to the target quality level of $q = 8.0$ was selected for comparison, which has the same en/decoder network architecture as that of SCR^{psnr}_{Hyp} and SCR^{psnr}_{MS}.

We perform decoding 100 times for each image in the Kodak image set [25] and measure the average decoding time of all the images. As shown in Fig. 9, SCR^{psnr}_{Hyp} and SCR^{psnr}_{MS} reduce the decoding time in the entire bit rate range compared with their variants without the selective compression. The decoding time savings are more noticeable for the low-quality compression owing to the reduction in the number of selected representation elements, and this is clearly seen when comparing the entropy decoding times (denoted in light blue color). In spite of the overheads from the reshaping (in dark blue) and 3D binary mask generation, the time saved in the entropy decoding process is definitely larger than the overhead time. It should be noted that the 3D binary mask generation time is included in the hyper decoder network time (gray) because the 3D binary mask generation shares most parts of the hyper decoder. SCR^{psnr}_{Hyp} and SCR^{psnr}_{MS} reduce the average decoding time by 11.28% and 8.32%, respectively, compared with their variants without the selective compression. Furthermore, SCR^{psnr}_{Hyp} and SCR^{psnr}_{MS} showed lower decoding times compared to the original reference compression models over all target quality levels except $q = 8$ of SCR^{psnr}_{MS}, in which SCR^{psnr}_{MS} yielded a slightly higher decoding time compared with the original Mean-scale [7] model.

5 Conclusion

We firstly proposed a 'selective compression of representations' (SCR) method for NN-based variable-rate image compression, which performs entropy coding only for the partially selected latent representations. The proposed SCR method selects essential representation elements for compression in an adaptive manner according to a given target quality level. For this, we first generate a 3D importance map, which represents the importance of each representation element independently of target quality levels, and then adjust it in a target quality-adaptive manner using the learned importance adjustment curves to generate a 3D binary mask that represents whether or not to select each representation element for compression. We demonstrated through experiments that the proposed SCR method could provide comparable coding efficiency to those of the separately trained reference compression models. Furthermore, the proposed SCR method showed less decoding time than those under comparison for almost all the cases. We also showed that the proposed SCR method could enable continuously variable-rate image compression through simple non-linear interpolation of the importance adjustment curves between discrete target quality levels in which the selective compression was trained.

References

[1] G. Toderici, S. M. O'Malley, S. J. Hwang, D. Vincent, D. Minnen, S. Baluja, M. Covell, and R. Sukthankar, "Variable rate image compression with recurrent neural networks," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. [Online]. Available: http://arxiv.org/abs/1511.06085

[2] N. Johnston, D. Vincent, D. Minnen, M. Covell, S. Singh, T. Chinen, S. Jin Hwang, J. Shor, and G. Toderici, "Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[3] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," in *the 5th Int. Conf. on Learning Representations*, 2017. [Online]. Available: http://arxiv.org/abs/1611.01704

[4] L. Theis, W. Shi, A. Cunningham, and F. Huszár, "Lossy image compression with compressive autoencoders," in *the 5th Int. Conf. on Learning Representations*, 2017. [Online]. Available: http://arxiv.org/abs/1703.00395

[5] O. Rippel and L. Bourdev, "Real-time adaptive image compression," in *International Conference on Machine Learning*, 2017. [Online]. Available: http://arxiv.org/abs/1705.05823

[6] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," in *the 6th Int. Conf. on Learning Representations*, 2018. [Online]. Available: http://arxiv.org/abs/1802.01436

[7] D. Minnen, J. Ballé, and G. Toderici, "Joint autoregressive and hierarchical priors for learned image compression," in *Advances in Neural Information Processing Systems*, May 2018.

[8] J. Lee, S. Cho, and S.-K. Beack, "Context-adaptive entropy model for end-to-end optimized image compression," in *the 7th Int. Conf. on Learning Representations*, May 2019.

[9] J. Zhou, "Multi-scale and context-adaptive entropy model for image compression," in *Workshop and Challenge on Learned Image Compression at CVPR*, Jun. 2019.

[10] T. Chen, H. Liu, Z. Ma, Q. Shen, X. Cao, and Y. Wang, "Neural image compression via non-local attention optimization and improved context modeling," *arXiv preprint arXiv:1910.06244*, 2019. [Online]. Available: https://arxiv.org/abs/1910.06244

[11] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Learned image compression with discretized gaussian mixture likelihoods and attention modules," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[12] J. Lee, S. Cho, and M. Kim, "An end-to-end joint learning scheme of image compression and quality enhancement with improved entropy minimization," *arXiv preprint arXiv:1912.12817*, 2019.

[13] D. Minnen and S. Singh, "Channel-wise autoregressive entropy models for learned image compression," in *The IEEE International Conference on Image Processing (ICIP)*, 2020.

[14] F. Bellard, "Bpg image format," 2014. [Online]. Available: http://bellard.org/bpg/

[15] D. S. Taubman and M. W. Marcellin, *JPEG 2000: Image Compression Fundamentals, Standards and Practice*. Norwell, MA, USA: Kluwer Academic Publishers, 2001.

[16] "Versatile video coding reference software version 7.1 (VTM-7.1)," December 2019. [Online]. Available: https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM/tags/VTM-7.1

[17] Y. Choi, M. El-Khamy, and J. Lee, "Variable rate deep image compression with a conditional autoencoder," in *International Conference on Computer Vision*, 2019.

[18] C. Cai, L. Chen, X. Zhang, and Z. Gao, "Efficient variable rate image compression with multi-scale decomposition network," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 12, pp. 3687–3700, 2019.

[19] Z. Cui, J. Wang, B. Bai, T. Guo, and Y. Feng, "G-vae: A continuously variable rate deep image compression framework," 2020. [Online]. Available: https://arxiv.org/abs/2003.02012

[20] O. Rippel, A. G. Anderson, K. Tatwawadi, S. Nair, C. Lytle, and L. Bourdev, "Elf-vc: Efficient learned flexible-rate video coding," in *International Conference on Computer Vision*, 2021.

[21] G. Lu, X. Zhang, W. Ouyang, L. Chen, Z. Gao, and D. Xu, "An end-to-end learning framework for video compression," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 10, pp. 3292–3308, 2021.

[22] M. Song, J. Choi, and B. Han, "Variable-rate deep image compression through spatially-adaptive feature transform," in *International Conference on Computer Vision*, 2021.

[23] T. Chen and Z. Ma, "Variable bitrate image compression with quality scaling factors," in *The IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020.

[24] Y. Lu, Y. Zhu, Y. Yang, A. Said, and T. S. Cohen, "Progressive neural image compression with nested quantization and latent ordering," in *The IEEE International Conference on Image Processing (ICIP)*, 2021.

[25] E. Kodak, "Kodak lossless true color image suite (photocd pcd0992)," 1993. [Online]. Available: http://r0k.us/graphics/kodak/

[26] P. Raiko, M. Berglund, G. Alain, and L. Dinh, "Techniques for learning binary stochastic feedforward neural networks," in *International Conference on Learning Representations (ICLR 2015)*, May 2015, pp. 1–10, international Conference on Learning Representations, ICLR ; Conference date: 07-05-2015 Through 09-05-2015.

[27] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *the 3rd Int. Conf. on Learning Representations*, 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[28] "Workshop and challenge on learned image compression," 2019. [Online]. Available: http://www.compression.cc/

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Sec. 4.1.
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We provide some essential source code blocks in Appendix A.1.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Sec. 3.3
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Sec. 4.2 and Fig. 9
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] We only used the assets open to the public.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Appendix

A.1 Example source codes for the representation selection and reshaping operators

In this section, we provide some essential source code blocks to implement the selection operator $M(\cdot)$ and the reshaping operator $Re(\cdot)$, as below:

```
import numpy as np
...
# representation selection operator M()
def M(rep, 3D_mask):
    mask_flattened = 3D_mask.ravel()
    idx_list = np.nonzero(mask_flattened)
    rep_flattened = rep.ravel()
    selected_rep = np.take(rep_flattened, idx_list)
    return selected_rep
...
# representation reshaping operator Re()
def Re(selected_rep, 3D_mask):
    reshaped_rep = np.zeros_like(3D_mask, dtype=float)
    reshaped_rep[3D_mask] = selected_rep
    return reshaped_rep
```

It should be noted that these two modules are used for the test phase. As described in Section 3.3, these two modules are not necessarily required for training.

A.2 Training details of the proposed SCR method

For more stable and faster training, we adopt a step-wise training including the following three steps: (i) In the first step, a fixed-rate compression model is trained for high quality compression corresponding to the target quality level $q = 8.0$ of a variable-rate model; (ii) In the second step, using the trained fixed-rate compression model as a pre-trained model, we train a SCR variant without the selective compression in an end-to-end manner; (iii) In the third step, using the trained SCR variant without the selective compression as a pre-trained model, we train the SCR full model in an end-to-end manner. We proceed with all the training steps until the performance of each step gets sufficiently converged, using the ADAM [27] optimizer. The numbers of training iterations for these three steps are $7M$, $1.2M$, and $1.2M$, respectively. As the training data set, we use $51,141$ 256×256-sized patches that were cropped, in a non-overlapping manner, from the whole CLIC [28] training set, and we set the batch size to 8. An initial learning rate is set to 5×10^{-5}, and a learning rate of 2×10^{-6} is used for the final $0.1M$ iterations. This learning rate decaying is conducted for all training phases.

A.3 Additional R-D curves

Fig. 10 shows the rate-distortion curves for the SCR^{ssim}_{Hyp} and SCR^{ssim}_{MS} models. It should be noted that we used the reproduced results for MS-SSIM-optimized reference compression models rather than the results reported in the original papers because the reproduced results showed substantially higher coding efficiency. The difference between the reproduced results and those of the original papers may be due to the different implementation of the MS-SSIM loss. On the other hand, in the case of the MSE-optimized reference compression models, the reproduced results were almost the same as those of the original paper, so we used the results in the original paper.

Fig. 11 shows the rate-distortion curves for SCR^{psnr}_{Cxt} and SCR^{ssim}_{Cxt}. Here, we also used the reproduced results of the MS-SSIM-optimized Context [7] model for the same reason above.

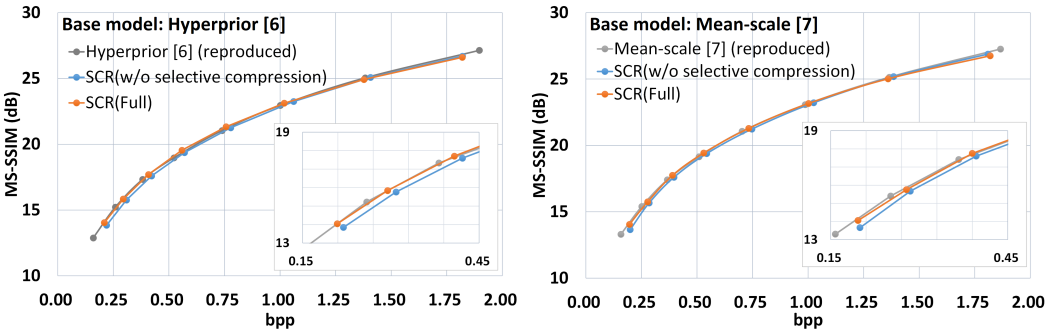


Figure 10: Rate-distortion curves of the separately trained reference compression model, the SCR model without the selective compression, and the SCR full model for discrete target quality levels. Here, MS-SSIM-optimized models are used. MS-SSIM values are converted into decibels($-10\log_{10}(1-\text{MS-SSIM})$) for visualization, in the same manner as in [6].

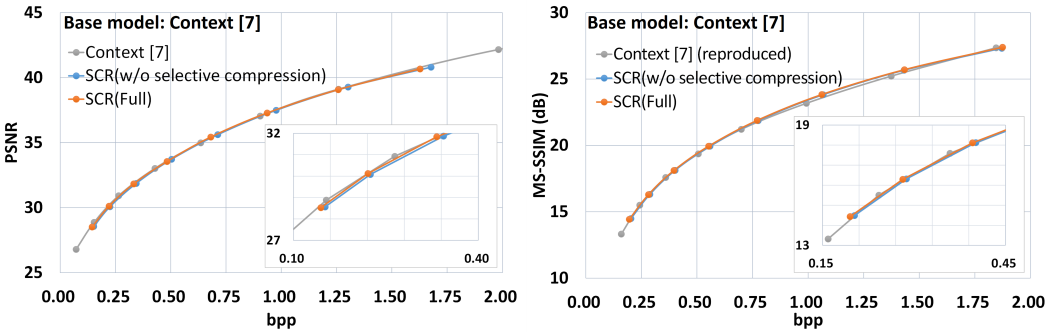


Figure 11: Rate-distortion curves of the separately trained reference compression model, the SCR model without the selective compression, and the SCR full model on the Context [7] base compression architecture, for discrete target quality levels. Left: the comparison results of the MSE-optimized models. Right: the comparison results of the MS-SSIM-optimized models

A.4 Reconstruction samples

In this section, we provide a few more supplemental reconstruction samples. Fig. 12, 13, and 14 show the cropped reconstructions of the KODIM04, KODIM20, KODIM23 images of the Kodak image set [25], respectively. Here, we used the SCR^{psnr}_{Hyp} model to generate the samples.

Figure 12: Cropped reconstruction samples of the KODIM04 image in the Kodak imgae set [25] (best viewed in digital format).The number embedded in each sample represents the target quality level q.

Figure 13: Cropped reconstruction samples of the KODIM20 image in the Kodak imgae set [25] (best viewed in digital format). The number embedded in each sample represents the target quality level q.

Figure 14: Cropped reconstruction samples of the KODIM23 image in the Kodak imgae set [25] (best viewed in digital format). The number embedded in each sample represents the target quality level q.