Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

journal homepage: www.elsevier.com/locate/jvci

Full length article

Deep image compression with multi-stage representation☆

Zixi Wang [a], Guiguang Ding [b], Jungong Han [c], Fan Li [a,*]

[a] School of Information and Communications Engineering, Xi'an Jiaotong University, Xi'an 710049, China
[b] School of Software, Tsinghua University, Beijing 100084, China
[c] Computer Science Department, Aberystwyth University, Aberystwyth SY23 3FL, UK

ARTICLE INFO

Keywords:
Deep image compression
Multi-stage representation
Data-dependent probability model
Convolutional neural network

ABSTRACT

While deep learning-based image compression methods have shown impressive coding performance, most existing methods are still in the mire of two limitations: (1) unpredictable compression efficiency gain when adopting convolutional neural networks with different depths, and (2) lack of an accurate model to estimate the entropy during the training process. To address these two problems, in this paper, a deep multi-stage representation based image compression (MSRIC) method is proposed. Owing to this architecture, the detail information of shallow stages and the compact information of deep stages can be utilized for image reconstruction. Furthermore, a data-dependent channel-wised factorized probability model (DCFPM) is adopted to increase the accuracy of entropy estimation. Experimental results indicate that the proposed method guarantees better perceptual performance at a wide range of bit-rates. Also, ablation studies are carried out to validate the above mentioned technologies.

1. Introduction

Nowadays the data volume of raw images collected by visual sensors becomes quite large, thus it is necessary to compress them in order to transmit and store images effectively. To this end, image compression has always been studied from a variety of disciplines including engineers, computer scientists, etc. A normal image compression framework is generally composed of three parts, i.e. transformation, quantization and entropy coding. Conventional image encoding standards such as JPEG [1], JPEG2000 [2] and HEVC intra-frame encoding (BPG) [3] generally adopt a pre-defined handcrafted transform, i.e. discrete cosine transform (DCT) or discrete wavelet transform (DWT), to reduce redundancy. Then, quantization and entropy coding are performed subsequently to obtain compressed bitstream. In the early years, researchers often focused on improving compression performance on the basis of existing conventional compression standards.

Recently, deep learning demonstrates exceptional performance in several computer vision tasks, also including image compression. Through neural network with proper architecture, raw image data can be transformed from the spatial domain into the feature domain, which is in some way similar to the frequency domain transformation in conventional image compression technology. Instead of the handcrafted designed transformation matrix, deep learning-based method is able to obtain more efficient transform model in a data-driven way. Therefore, the compression efficiency can be improved. Several deep image compression methods have been studied during the past years. By the design of differentiable quantization, deep image compression can be modeled as an end-to-end training architecture, as firstly presented in [4]. After that, studies about deep image compression mainly focus on two directions: (1) mining better network architecture to improve compression efficiency and (2) maximizing the compaction of actual latent representation once the network architecture is determined. More details of related works will be discussed in Section 2. In this paper, we focus on both aspects to further improve deep image compression method.

With respect to the first aspect, we notice that previous studies often design a straight-forward network architecture and then encode the output feature maps of the last layer into bitstream. A general architecture is often composed of three down-sampling operations and several convolutional layers in between. However, the depth of the network varies among these studies. For example, three convolutional layers are adopted in order to compress the raw image data in [4]. In [5], three residual blocks are utilized, where each block consists of an additional two convolutional layers. In [6], a similar network architecture as [5] is proposed but with 15 residual blocks. Intuitively, a deeper network is beneficial for extracting more compact representation, thus the network tends to be deeper in order to obtain a more efficient transform model. However, growing network depth is at risk of losing more information in texture detail. From this perspective, the

☆ This paper has been recommended for acceptance by Zicheng Liu.
* Corresponding author.
E-mail address: lifan@mail.xjtu.edu.cn (F. Li).

https://doi.org/10.1016/j.jvcir.2021.103226
Received 14 July 2020; Received in revised form 21 April 2021; Accepted 10 July 2021
Available online 21 July 2021

network should be shallower in order to reduce the information lost during compression and obtain better reconstruction performance. In light of the above two points, both shallow and deep network seem to have their own superiorities, thus, it is a meaningful study to find an architecture that can combine these advantages.

Regarding the second aspect, an accurate probability model of latent representation is quite important during the training process so as to estimate actual bit-rate correctly, and thus has a direct influence on the final compaction of output feature maps. A handy solution as proposed in [4] is to formulate a piece-wise linear function to approximate the distribution of extracted latent representation. After that, hyperprior [7] is proposed to obtain more accurate approximation. Despite impressive performance, a hyperprior module is generally composed of several convolutional layers with down-sampling and up-sampling, which involves more complexity and may limit its application in practical scenarios. Considering the complexity burden, we follow and improve the previous implementation to approximate the distribution by a piecewise linear function. We notice that the distribution of latent representation will be different once the input image changes. Thus, to decrease the margin between the estimated probability model and actual distribution, the probability model of latent representation should be data-dependent.

To address the above-mentioned problems, a deep multi-stage representation based image compression (MSRIC) method is proposed in this paper. Besides, a data-dependent channel-wised factorized probability model (DCFPM) is implemented during the training process. Our main contributions are summarized as follows:

- A deep image compression model with multi-stage representation is presented. Here, a novel multi-stage network framework is specifically designed such that the latent representation is composed of multi-stage feature maps. Thanks to this architecture, both textural information extracted by shallower layers and abstract information extracted by deeper layers coexist in a compressed bitstream, and multi-stage representation can be simultaneously utilized to synthesize the reconstructed image. In this way, the advantages of shallow network and deep network are combined, thus a more compact transform can be learned and the compression efficiency is improved.
- A data-dependent channel-wised factorized probability model (DCFPM) for latent representation is proposed. We argue that the distribution of latent representation is discrepant among different input images and different output channels. To this end, separate probability models are modeled for each output channel of each input image in a batch. In this way, estimating compressed bit-rate during the training process can be more accurate, thereby improving the final compression performance.
- Experimental results turn out that the proposed MSRIC can achieve better compression efficiency over a large rate scales, compared with BPG, when being evaluated by multi-scale structural similarity index (MS-SSIM) metrics. Moreover, ablation study demonstrates the performance improvement if introducing multi-stage representation and data-dependent probability model.

The remainder of this paper is organized as follows. Section 2 reviews the existing studies of deep image compression. Section 3 presents the framework of MSRIC in details. Section 4 gives the experimental results. Section 5 concludes this paper.

2. Related work

In this section, previous works about deep image compression are briefly reviewed in two categories: (1) mining more efficient network architecture to increase the efficiency of extracting representation and (2) trying to maximize the compaction of actual latent representation so as to minimize the coding rate.

For the first category, several advanced network frameworks are introduced in deep image compression task, including convolutional auto-encoders (CAEs), recurrent neural networks (RNNs), generative adversarial networks (GANs), and others. Ballé et al. [4] firstly modeled deep image compression as a variational auto-encoder with convolution layers and incorporated differentiable approximations of quantization so that the network can be trained end-to-end by gradient backpropagation. Moreover, fully factorized prior probability model was proposed to estimate the entropy of latent representation. Theis et al. [5] introduced residual module into auto-encoder architecture. And the network was furthermore suitable for high-resolution images thanks to sub-pixel architectures. Toderici et al. [8] presented a long short-term memory (LSTM) recurrent network to compress low-bytecount image previews (thumbnails). This study was further extended to full resolution images in [9], and the performance was able to outperform JPEG at most bit-rates. In [10], Johnston et al. proposed a recurrent architecture and a spatially adaptive bit allocation algorithm. In [11], Agustsson et al. proposed a GAN-based extreme learned image compression architecture. Akbari et al. [12] adopted residual coding framework and the semantic segmentation map of the input image was also utilized to reconstruct the image. In [13], Huang et al. proposed a multi-scale framework and also employed GANs with multi-scale discriminators. Compared with other studies, better visually pleasing images could be obtained at significant lower bit-rates by GANs. However, despite better subjective quality results at extreme low bit-rates, GAN-based methods show worse performance at high bit-rates, which is also important for image compression task. Moreover, according to existing studies, CAE-based methods typically present better RD performance than RNN-based methods. Therefore, we propose a CAE-based architecture in this paper.

For the second category, researchers generally make effort to improve the accuracy of entropy model during the training process, so the compaction of actual latent representation can be increased during the real coding process. Ballé et al. [7] firstly introduced a hyperprior to approximate the actual distribution of latent representation. In this way, the spatial dependencies in the latent representation were captured so that the accuracy of the entropy model was improved. Moreover, the side information generated by hyperprior was also beneficial to entropy coding. Zhou et al. [14] modeled the prior probability of compressed representation as a Laplacian distribution. What is more, a post-processing module was proposed in order to remove the compression artifacts and blurs for low bit-rate images. A rate control algorithm was also applied to further improve coding efficiency. Liu et al. [15] jointly took the hyperpriors and autoregressive priors for conditional probability estimation and proposed a non-local module to capture global correlations of input images so as to improve compression efficiency. Cheng et al. [16] used discretized Gaussian Mixture Likelihoods to parameterize the distributions of latent codes. Attention modules were also incorporated into network architecture to enhance the performance.

There were also several studies that focus on other techniques to improve the compaction of latent representation. In [17], Cheng et al. designed a CAE architecture to extract compact representation. Then principal components analysis (PCA) was utilized to generate a more energy-compact representation. In their follow-up work [18], they provided a mathematical analysis on the energy compaction property for CAE and proposed a normalized coding gain metric in neural networks, which could act as a measurement of compression capability. Campos et al. [19] proposed a content adaptive optimization that optimized the latent representation individually. Note that an iteration procedure was adopted during encoding, while the computation on the decoder remains unchanged. Li et al. [20] took the spatial variation of image content into account and designed an importance map subnet to produce the importance mask for locally adaptive bit-rate allocation. Moreover, the entropy of latent representation could be estimated by summation of importance map.

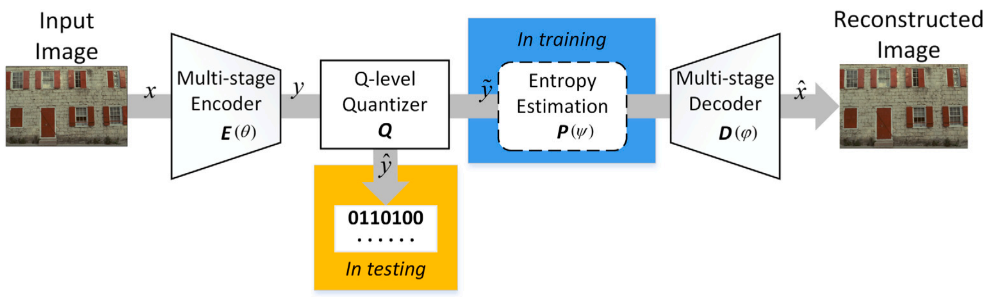

Fig. 1. Overall Framework of MSRIC.

Nevertheless, the above-mentioned methods generally have two shortcomings, i.e. additional computational complexity and extra side information, which may limit its application in practical scenes. Specifically, a hyperprior module is actually an additional convolutional network besides the ordinary image compression auto-encoder, as in [7, 14–16]. Introducing the hyperprior module not only increases the training difficulty, but also improves the computational burden in practical applications. Also, both the hyperprior module and importance map subnet [20] lead to extra side information to be stored in a compressed bitstream. It may limit the compression ratio.

In this paper, we adopt a data-dependent channel-wised factorized probability model to approximate the entropy of latent representation during the training process. There is no additional convolutional layer apart from auto-encoder to be taken into account and no extra side information need to be considered in the compressed bitstream. This probability model leads to better training efficiency and higher compression performance of the proposed network.

3. Deep multi-stage representation based image compression

In this section, we present the details of MSRIC, including a multi-stage encoder network $E(\theta)$, a multi-stage decoder network $D(\varphi)$, a Q-level quantizer Q, and a probability model $P(\psi)$ for entropy estimation. The overall framework of MSRIC is depicted as Fig. 1. Specifically, the architecture of deep multi-stage image encoder $E(\theta)$ and decoder $D(\varphi)$ with trainable parameter sets θ and φ are firstly presented. Then the quantization rule of Q is introduced. The approximation of quantization during the training process is determined, so that the auto-encoder can be end-to-end trained. Finally, the data-dependent channel-wised factorized probability model $P(\psi)$ with trainable parameter set ψ is presented to estimate the entropy during the training process.

3.1. Multi-stage encoder and decoder networks

To improve the compression efficiency of auto-encoder, a multi-stage encoder and network framework is proposed in this section.

Existing CAE-based image compression methods generally design a straight-forward network architecture. Then the output feature maps of the last convolutional layer are regarded as latent representation and encoded into bitstream. However, the containing information of feature maps varies as the network depth changes according to existing knowledge about deep learning. Specifically, for image compression task, the extracted latent representation should keep the information of the original image as much as possible. To this end, the network has a tendency to be shallow. This is because that the loss of pixel-level detail information will increase, when the network becomes deeper with more convolutions and nonlinearities. On the contrary, a deeper network is more beneficial for extracting the most important global information of the original image, since the range of receptive field expands as number of convolution grows. It will be helpful for obtaining more compact representation and improving the compression efficiency. Therefore, how to determine the network depth of auto-encoder is always an open question, since feature maps extracted by both deep and shallow network have a specific contribution to improving compression efficiency.

From the above, a multi-stage framework is proposed for our deep image compression encoder and decoder network. In this way, the extracted feature maps combine the attributions of both deep and shallow network. Therefore, the containing information in bitstream becomes richer, which is helpful for reconstructing the original image. Thanks to this architecture, the compression performance can be improved.

The architecture diagrams of deep multi-stage deep encoder is illustrated as Fig. 2. For an input image $x \in R^{H\times W\times 3}$ in 8-bit RGB format with height H and width W, multi-stage latent representation is extracted by the encoder network $E(\theta)$, as follow.

$$E(\theta) : x \mapsto \left\{Rp_i\right\}_{i=1}^{4} \tag{1}$$

where Rp_i denotes the latent representation extracted by the ith stage. There are totally 4 stages in our network architecture. Due to this framework, both the local and fine information from shallow layers (i.e. stage 1 and stage 2) and the global compact information from deep layers (i.e. stage 3 and stage 4) can be stored in the coded bitstream. The details of the proposed model are presented below.

The value of raw image data is firstly normalized to $[0,1]$. After that, a convolutional layer is adopted to expand the number of image channel from 3 to 192. Then 8 residual units and 3 down-sampling layers following by generalized divisive normalization (GDN) [4] nonlinearities form the main stream of encoder network. Note that the down sampling layer is implemented by an ordinary convolutional layer with a stride of 2. Since there are 3 down-sampling layers, the final output feature maps of the last stage have a size of $H/8 \times W/8$.

To obtain multi-stage representations, the network is manually divided into 4 stages separated by GDN activations. Once a stage is finished, the output feature maps are extracted to form the corresponding latent representations. Down-sampling layers are adopted for the feature maps extracted by stage 1 and stage 2, to guarantee compression efficiency as well as ensure the extracted latent representations are of the same size and can be concatenated with those extracted by other stages. The filter numbers of all convolutional layers, including those in residual units, are set to 192 based on experimental results. The first and the last convolutional layers adopt 5×5 convolution, and 3×3 convolution is adopted in all other convolution layers. It is worth-noting that sigmoid activation is applied as output activation so that the value of latent representation is in a range of (0,1). The number of output channels of each stage are reduced from 192 to 16 by the last convolution operations before sigmoid activation. The total channels of concatenated feature maps are 64. Specifically, the output feature maps of each stage have a size of $H/8 \times W/8 \times 16$, and the concatenated latent representations have a size of $H/8\times W/8\times 64$. Once the latent representations are obtained, quantizer and entropy coder are implemented to further compress them into bitstream.

To intuitively understand the containing information of each stage, we visualized total 64 output latent representation maps of a test

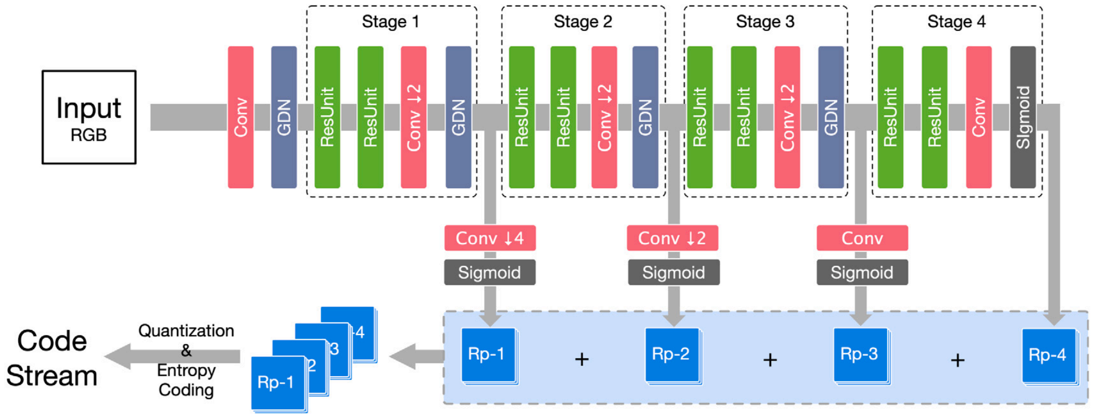

Fig. 2. Illustration of the deep multi-stage deep encoder. The local and fine information is extracted by shallow stages (i.e. stage 1 and 2). The global and compact information is extracted by deep stages (i.e. stage 3 and 4).

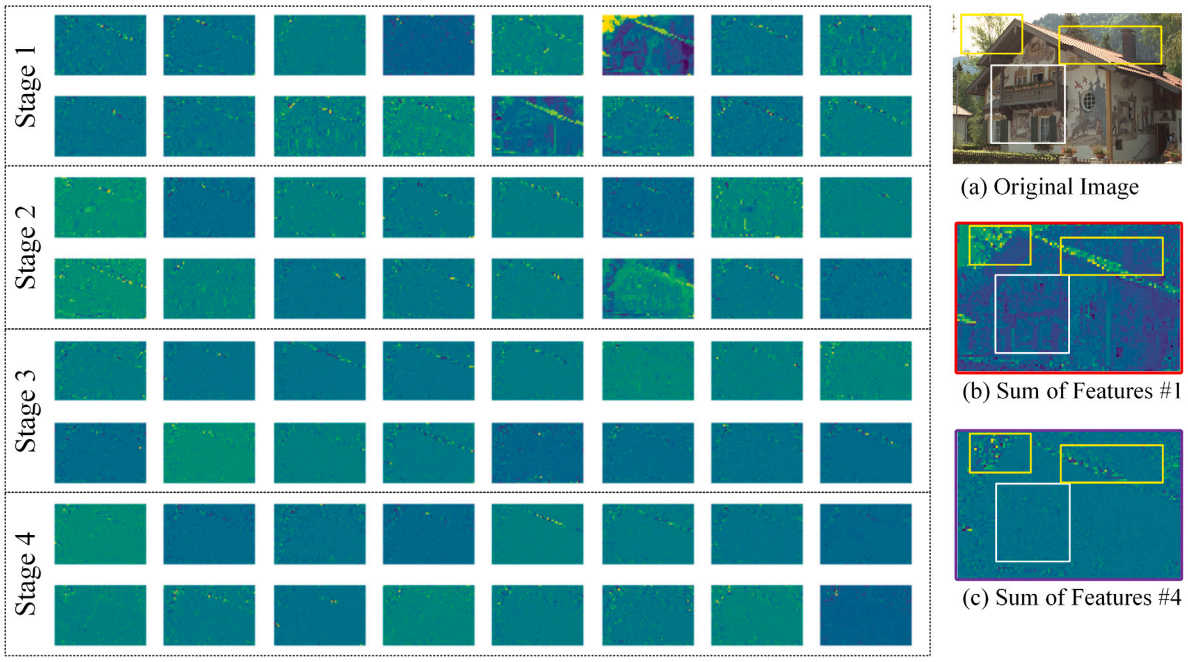

Fig. 3. Visualization of multi-stage latent representation. From top to bottom at left part of this figure, each two rows present 16 output feature maps extracted by a specific stage. Total 64 feature maps of 4 stages corresponding to those in Fig. 2 are illustrated. At the right part, original image and the visualization for sum of features in stage 1 and 4 are presented.

image as an example. As presented in Fig. 3, the extracted feature maps from different stages contain diverse information. In general, the local and detail texture information is stored in those feature maps extracted by stage 1 and stage 2. As the network depth grows, the detail information loss increases. For those representations extracted by stage 3 and stage 4, it can be noticed that the content of feature maps become hard to be understood by humans. It is in line with our knowledge that the extracted feature will be more abstract and compact when a deeper network is adopted. The right part of this figure provide the visualization of original image and sum of features for stage 1 and 4. With stage 1, the general appearance and details of the original image remains. It still provides readable content for humans. This result implies that the compression efficiency of stage 1 could be low since the sparsity of feature maps is quite limited. With stage 4, the visualization for sum of features indicates the loss of detail information. We notice that the textural pattern of this house is nearly invisible in this visualization (white squares). However, the distinct structural patterns in original image still remain, such as the lines of roof (yellow squares), and the compaction of features improves. Since the dissimilarity of representation from different stages, we argue that a multi-stage framework will be beneficial for improving the richness of extracted information in latent representation. Therefore, the compression efficiency will be improved for the proposed auto-encoder.

For the decoder side, the extracted multi-stage representation is utilized to synthesize the reconstructed image as follow.

$$D(\varphi) : \left\{Rp_i\right\}_{i=1}^{4} \mapsto \hat{x} \tag{2}$$

As indicated in Fig. 4, the compressed bitstream is firstly decoded by entropy decoder. A symmetric architecture with encoder network is specifically proposed for decoder network. The latent representations from different stages are fed into the main stream of decoder network exactly according to the stages from which they are extracted. We argue that this symmetric architecture is beneficial to restore the image from the extracted multi-stage representations.

For each stage, convolutional layers are firstly applied, in order to expand the channel of input feature maps from 16 to 192. Then pixel shuffle layers are adopted for stage 1 and stage 2 to up sample these features maps, to make sure that they are in the same size of main

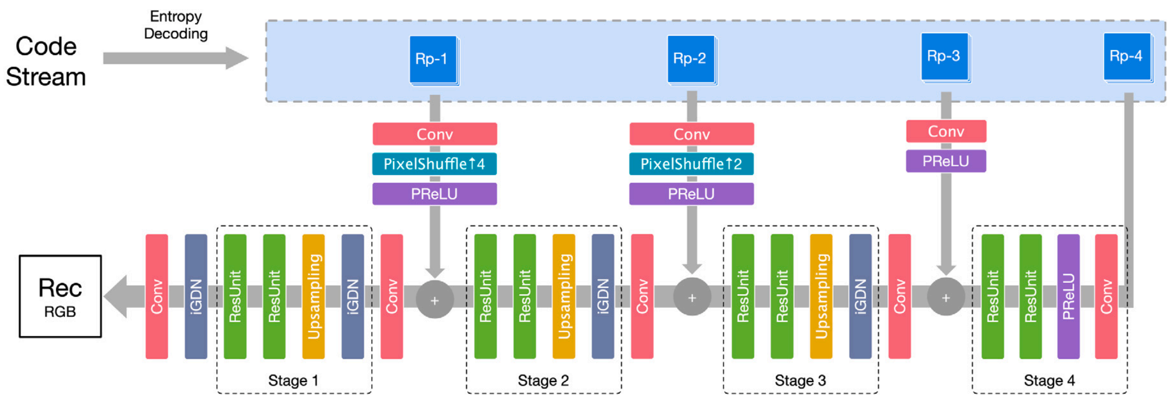

Fig. 4. Illustration of the deep multi-stage deep decoder. The network adopts a symmetric architecture as the encoder network. The latent representations are fed into the decoder network exactly according to the stages from which they are extracted. Here "+" does not refer to addition of pixel value, but to concatenation of two sets of feature maps.

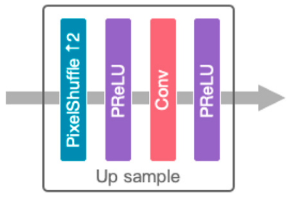

Fig. 5. Up-sampling module in the multi-stage deep decoder network.

stream features so that can be concatenated with them. After the up-sampled feature maps are fed into the main stream of decoder network, following convolutional layers are adopted to fuse and concentrated the concatenated features back into 192 channel. Therefore, the rest part of decoder network is conducted in a consistent convolutional width of 192.

For the main stream of the network, the up-sampling module is achieved by a series connection of pixel shuffle layer and convolutional layer, as depicted in Fig. 5. To guarantee symmetry, the GDN nonlinearities are replaced by iGDN nonlinearities in decoder network. Prelu layers are adopted to increase nonlinearity. Then the reconstructed image is mapped back to $[0, 255]$ at the end of the network.

3.2. Q-level quantization

Since the pixel values of output feature maps are continuous values in a range of $(0, 1)$, quantization operation is required to discretize them so that the compressed bit-rate can be controlled. Assuming that quantization level is denoted by Q and y is the pixel value of original extracted latent representation. According to Section 3.1, y is a continuous value in a range of $(0, 1)$, then the quantized discrete value $\hat{y}$ can be obtained by

$$\hat{y} = round\left[y \cdot \left(2^Q - 1\right)\right] \tag{3}$$

By adopting this quantization rule, the original value is firstly scaled from $(0, 1)$ to $\left(0, 2^Q - 1\right)$, then the scaled value is rounded to the nearest integer. Thus, the quantized representations have discrete value in a range of $\left[0, 2^Q - 1\right]$. However, it is worth noting that the derivatives of the round quantization function are zero almost everywhere, resulting in the ineffective of gradient descent. Following [4], the round function is replaced with an additive uniform noise during the training process,

$$\tilde{y} = y \cdot \left(2^Q - 1\right) + \varepsilon \tag{4}$$

where ε is an addictive i.i.d. uniform noise, which is frequently used as a model of quantization error. Due to this approximation, stochastic gradient descent optimization can be adopted and the auto-encoder network can be end-to-end trained. In this paper, the quantization level is set to 6 based on experiments.

3.3. Data-dependent channel-wised factorized probability model

To maximize the training performance of our model, a data-dependent channel-wised factorized probability model (DCFPM) is presented in this section.

For a deep lossy image compression auto-encoder, both distortion level and rate consumption are necessary evaluation indexes. Therefore, the loss function for training is defined as a weighted combination of these two terms, as follows:

$$L = D_x + \lambda \cdot R \tag{5}$$

where λ can be adjusted to balance the tradeoff between these two parts of losses. In our model, D_x is measured by mean square error (MSE) matrix between original image and reconstructed image. By this setting, the rate-PSNR performance of our model can be maximized.

As presented in Section 3.2, the quantized value of latent representation is replaced by an approximation, so that the end-to-end training can be implemented for auto-encoder. Due to this approximation, the quantized value of latent representation is unknown during the training process. Therefore, the real bit-rate of latent representation after entropy coding cannot be directly calculated during the end-to-end training. To this end, the estimation of entropy is a necessary procedure in training a CAE. Following the implementation in [4], we calculate the entropy of approximated latent representation to estimate the real bit-rate, since the actual rate achieved by entropy encoder is only slightly larger than the entropy. It is a common approximation in other studies about CAE.

We argue that the accuracy of estimated entropy directly influences the training performance of the network model, thus effecting the final coding efficiency. Considering the diversity of probability distribution of different feature maps extracted by different input image, a data-dependent channel-wised factorized probability model is proposed to estimate the entropy of coded bitstream, during the training process. The detail of the proposed model is as follows.

Since the quantized value of continuous latent representation is approximated by $\tilde{y}$, the differential entropy of $\tilde{y}$ can also be used as an approximation of the entropy of $\hat{y}$. Let an element with index i of feature map with index j be denoted by $y_{j,i}$, then the approximation of quantized value is denoted by $\tilde{y}_{j,i}$. Therefore, the total information entropy I of latent representation extracted by one training image can be calculated by,

$$I = E_{\tilde{y}_{j,i} \sim P\left(\tilde{y}_{j,i}\right)}\left[-\log_2 P\left(\tilde{y}_{j,i}\right)\right] \tag{6}$$

Note that $P\left(\tilde{y}_{j,i}\right)$ is unknown. In order to obtain the value of $P\left(\tilde{y}_{j,i}\right)$, finely sampled piecewise linear functions are utilized to fit the probability density function curve of $P(\tilde{y})$. Specifically, we argue that the pixels belong to the same channel of the latent representation obeys one single probability distribution, which is so-called "channel-wised". What is more, once the input training image is changed, the distribution should also be modified. That is, the probability density function should be data-dependent. Therefore, a factorized probability model DCFPM that combines data-dependent and channel-wised attributions is proposed to achieve accurate estimation of entropy for latent representation.

Assuming that the batch size of the mini-batch gradient descent algorithm is B. For a batch b, let the current input training image be denoted by x_b. There are total of 64 sampled piecewise linear functions adopted to fit the distribution of feature maps. A single linear function corresponds to a single channel. Then, the parameters of the piece-wise linear function of batch b can be written as $\left\{\psi_c \left| x = x_b \right.\right\}_{c=1}^{64}$. Due to the data-dependent assumption, the piecewise linear functions of the same channel while belong to different input images should be updated separately. Therefore, the total parameters needed to be updated in one training iteration is denoted by $\psi = \left[\left\{\psi_c \left| x = x_b \right.\right\}_{c=1}^{64}\right]_{b=1}^{B}$. According to the above, there are $64 \times B$ sampled piecewise linear functions in total to be optimized during training process, so that the value of $P\left(\tilde{y}_{j,i}\right)$ can be obtained as accurate as possible. Specifically, the piecewise linear functions are updated similarly to one-dimensional histograms. For each batch b, the difference D_f between estimated value $\hat{P}\left(\tilde{y}_{j,i}\right)$ calculated by piecewise linear functions and the parameter vector of ψ_c is minimized by gradient descent. Thus the parameters of ψ tend to consist of the value of $P\left(\tilde{y}_{j,i}\right)$, and will be updated during each iteration.

4. Experiments

In this section, the performance of the proposed MSRIC is compared with existing conventional image compression standards and other deep image compression models. Then ablation studies are given in order to assess the validity of the multi-stage framework and DCFPM.

4.1. Experimental setup

We used a subset of ImageNet database [21] as the training dataset, following the setting in [4]. There were 8706 images in total, which then are cropped into 128 × 128 samples with random flips. Also, the training samples were saved as lossless PNG format so as to avoid compression artifacts.

During training process, two separate optimizers was employed to update different parameter sets. To update the parameters of auto-encoder (i.e. θ and φ), Adam optimizer [22] was adopted with a batch size of 10. The learning rate of Adam was fixed to 5×10^{-5} during the training process, since the learning rate decay strategy has no benefit for performance improvement according to experiments. Moreover, stochastic gradient descent was utilized to update the parameters of piecewise linear functions (i.e. ψ). To reduce the calculation burden, 2 sampling points were used for each unit interval of piecewise linear functions. To conclude, the entire end-to-end training process of MSRIC is described in Algorithm 1.

During training, we firstly trained an initializing model without rate restriction. That is, the value of λ in loss function was set to zero at the beginning of training. After enough iterations, the best potential performance could be obtained for the proposed framework. Then models with λ in range $[0, 0.004]$, with an interval of 0.0002, were trained. For these models, the network parameters were initialized by the pre-trained model and then were fine-tuned according to the corresponding loss functions with different λ.

For performance evaluation, the widely-used Kodak PhotoCD lossless dataset with 24 uncompressed 768×512 or 512×768 images was adopted. The compression rate was evaluated by bits per pixel (bpp), which is defined as the total amount of bits of the compressed bitstream

Algorithm 1 End-to-end Training for MSRIC

```
1:  while loop in MAX iterations do
2:      Input training image $x$
3:      $y := E(x;\theta)$
4:      $\tilde{y} := y \cdot \left(2^Q - 1\right) + \varepsilon$
5:      $\hat{x} := D\left(\tilde{y}/\left(2^Q - 1\right);\varphi\right)$
6:      Calculate $D_f$
7:      Update $\psi$ by $SGD$
8:      Calculate $L = D_x + \lambda \cdot R$
9:      Update $\theta$ and $\varphi$ by $Adam$
10: return
```

divided by the number of pixels. The reconstruction performance was evaluated by the peak signal-to-noise ratio (PSNR) and multi-scale structural similarity (MS-SSIM). MS-SSIM (dB) is $-10\log_{10}(1 - ms\_ssim)$. Therein, PSNR matrix assessed the pixel-level similarity between the original image and the reconstructed image, while MS-SSIM matrix assessed the perceptual performance.

4.2. Network parameter selection

To compare the performance of different network parameters and validate the efficiency of MSRIC, we conducted experiments based on different parameter settings, by modifying *the number of convolution channels*, *the number of intermediate latent representation channels* and *the quantization level*. The network architecture with 192 channels of convolution, 64 channels of latent representation and 6-level quantization is used as a baseline for comparison. The training settings are exactly same to those in Section 4.1 for fair comparison.

(1) Effect of Different Numbers of Convolution Channels: We trained three sets of models when the numbers of convolution channels are 128, 192 and 256 respectively. According to Fig. 6(a), there is indeed a correlation between number of convolution channels and compression efficiency. A higher number of channel increases the capacity of network so that improves the ability to approximate the optimal transformation from pixel domain to feature domain. However, the performance gain saturates as the number of channels rising. When increasing the channel number from 128 to 192, there is distinct RD performance improvement among a large bit-rate range; while when regarding 192 to 256, the improvement becomes tiny and even negligible at lower bit-rates. Therefore, we set the number of convolution channels as 192 in MSRIC.

(2) Effect of Different Numbers of Intermediate Latent Representation Channels: Models with different number of intermediate latent representation channels were trained to validate the effect of different settings. The numbers of channels for each stage were set to 8, 16 and 32, so that the total numbers of channels were 32, 64 and 128 respectively. As shown in Fig. 6(b), the RD performance degrades drastically when the numbers of channels for each stage were set to 8. This is because the upper limitation of reserved information in compressed stream is restricted on account of the size of feature maps, thus the RD performance will be limited. Similar to the effect of different numbers of convolution channels, the performance will also saturate as the latent representation goes wider. Besides, a larger number of representation channels leads to more data to be encoded by entropy coder, which may result in higher bit-rate of compressed stream and have impact on compression efficiency. Therefore, we select 16 channels for each stage as latent representation.

(3) Effect of Different Quantization Levels: To evaluate the effect of quantization level, the models with quantization levels 5, 6 and 7 were trained respectively. As presented in Fig. 6(c), the models with quantization level of 5 have intuitively worse performance than those with quantization level of 6. The impact of quantization error accounts for this. However, when increasing the level from 6 to 7, the

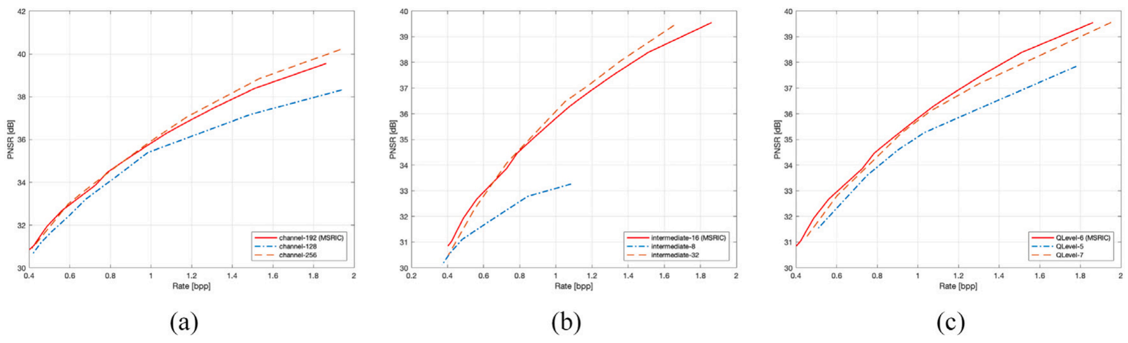

Fig. 6. Rate-Distortion (PSNR) curves with different network parameters. (a) with different numbers of convolution channels. (b) with different numbers of intermediate latent representation channels. (c) with different quantization levels.

RD performance decreases. We argue that this is because increasing the quantization level introducing more potential code words types to be encoded. Despite of lower quantization error resulting from better reconstruction performance, more code words types lead to higher bit-rate and do more harm to the RD performance. Therefore, 6-level quantization is employed in MSRIC.

4.3. Compression efficiency performance

To evaluate the compression efficiency of our proposed MSRIC, the rate–distortion (RD) curve are presented. To fairly evaluate our proposed method, both conventional image compression standards and other deep image compression method were compared. For conventional coders, JPEG [1], JPEG 2000 [2], JPEG XR [23], BPG [3] and versatile video coding (VVC) were considered. Libjpeg [24] was adopted to evaluate JPEG standard. OpenJPEG [25] was adopted to evaluate JPEG 2000 standard. JPEG XR Reference Codec [26] was adopted to evaluate JPEG XR standard. For BPG standard, libbpg [27] was adopted with the 4:2:0 chroma format. The VVC Test Model (VTM) [28] was adopted to evaluate VVC. In terms of deep image compression methods, we reproduced the network architecture of [4] with their provided code [29]. For a fair comparison, we trained the proposed network of [4] and [18] based on our training database. We exactly followed the training settings presented in these papers. We also evaluated the performance of [16] using the CompressAI Library [30]. Note that for the above learning-based approaches, we evaluated the MSE optimized models, so that the training optimization objectives of them and ours could be consistent. Moreover, since the source codes of other works are generally not available, we carefully digitalized the RD curves of [5,10] and [31] in our comparison experiments. The PSNR curves are not provided in [10] and [31], so that only MS-SSIM curves of them are presented as the comparison result.

The comparison results of RD curves are presented in Figs. 7 and 8. The results are average performance among Kodak PhotoCD database. Compared with conventional compression standards, our proposed MSRIC outperforms JPEG2000 in terms of PSNR performance and outperforms BPG in terms of MS-SSIM performance. At the bit-rates higher than 1 bpp, MSRIC even outperforms VVC with regards to MS-SSIM. Compared with other deep image compression algorithms, ours achieves higher coding efficiency than [4,5] and [18] at a large range of bit-rate, both on PSNR and MS-SSIM performance. Better MS-SSIM performance is also presented when compared with [10] and [31]. Although [16] performs better RD performance than ours, we argue that their approach introduces more complexity, by utilizing an additional convolution network to learn the parameters of Gaussian mixture model for each element and encoding the side-information by another entropy coder. High complexity may limit the application of an image coding scheme. More details will be discussed in the following evaluation on complexity performance.

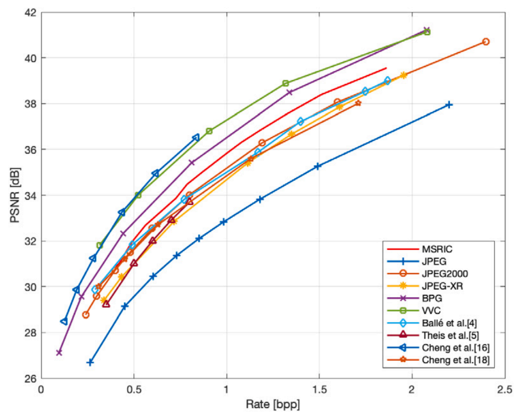

Fig. 7. Rate-Distortion (PSNR) curves of different compression methods.

We notice that deep learning-based image compression method often shows better perceptual performance (i.e. MS-SSIM) than pixel-level similarity performance (i.e. PSNR). In terms of MSRIC, higher MS-SSIM and lower PNSR performance can be obtained at the same time when compared with BPG. We argue that it is because the loss of detail texture information in pixel level is hard to avoid, due to the convolve operations in a neural network. However, it can be observed that MSRIC could achieve better PSNR performance than other deep image compression methods that both use factorized entropy model and do not introduce additional estimation network [4,5,10,18,31]. We attribute this improvement to our multi-stage framework. As discussed in Section 3, both local detail information extracted by shallow layers and global structure information extracted by deep layers can contribute to the synthesis of the reconstructed image. For the shallower layers, less convolve operations are carried out, thus more pixel-level detail information remains. Therefore, MSRIC can achieve better pixel-level similarity between the original image and the reconstructed image.

Besides, the compression efficiency varies with different bit-rates. For both PSNR and MS-SSIM performance, we observe that the improvement of MSRIC is more noticeable at higher bit-rates. Specifically, the PNSR performance is improved by about 0.5 dB when the coded bit-rate is larger than 0.8 bpp, using JPEG2000 as a benchmark. Then the improvement gradually decreases as bit-rate degrades. When bit-rate is smaller than 0.5 bpp, the PSNR performance of JPEG2000 matches ours. Similarly, the MS-SSIM performance of MSRIC is 1 dB higher than BPG when bit-rate is larger than 1.0 bpp. While the improvement is negligible when bit-rate is smaller than 0.4 bpp. This is because compression at higher bit-rates requires greater approximation capacity. In other words, a higher bit-rate leads to a more complex transform set for compressing an image. Therefore, the superiority of our multi-stage framework can be maximized at higher bit-rates. In contrast, the

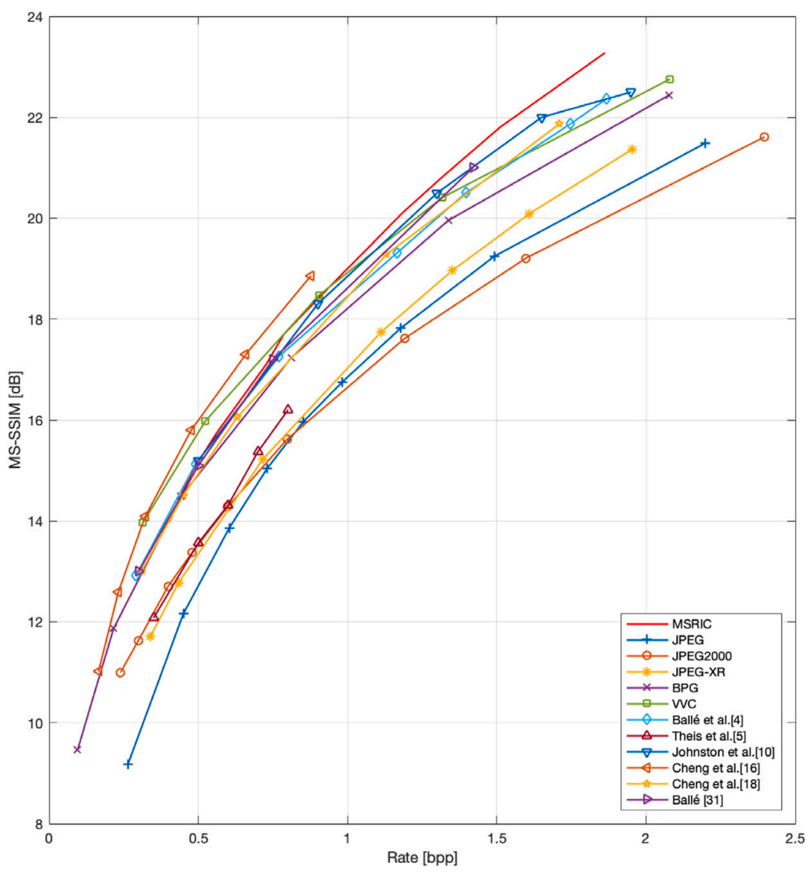

Fig. 8. Rate-Distortion (MS-SSIM) curves of different compression methods.

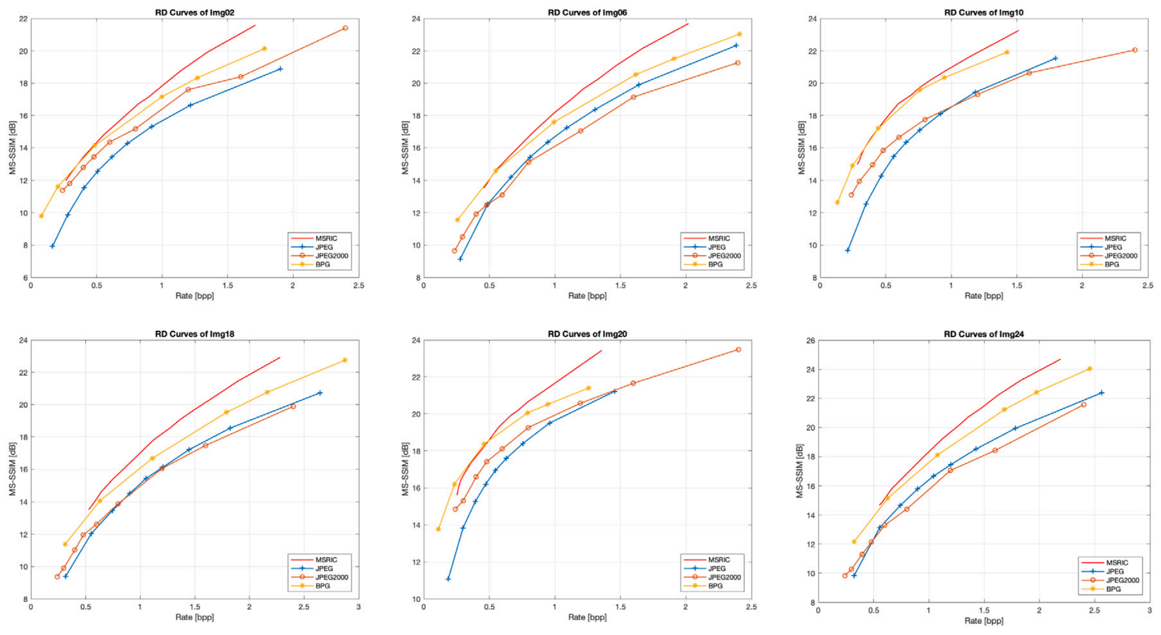

Fig. 9. Rate-Distortion (MS-SSIM) curves for single images.

approximation capacity of all models is sufficient at lower bit-rates, due to the rate limitation. Therefore, the improvement of MSRIC degrades as rate decreases. The discussion in [31] also confirms these results.

The RD curves for single images in Kodak dataset are also presented (Fig. 9). The six test images are randomly selected with various contents. According to the illustration, the compression efficiency improvements are mainly consistent among different inputs. Therefore, the conclusion based on average performance that we discussed before still holds for single images.

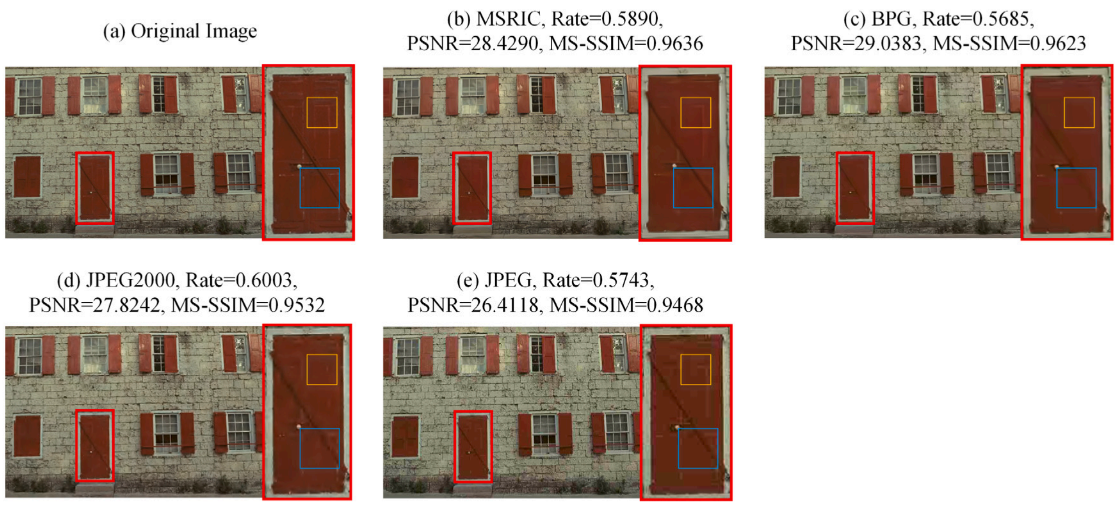
(a) Original Image
(b) MSRIC, Rate=0.5890, PSNR=28.4290, MS-SSIM=0.9636
(c) BPG, Rate=0.5685, PSNR=29.0383, MS-SSIM=0.9623
(d) JPEG2000, Rate=0.6003, PSNR=27.8242, MS-SSIM=0.9532
(e) JPEG, Rate=0.5743, PSNR=26.4118, MS-SSIM=0.9468

Fig. 10. Example of one test image in Kodak dataset compressed by different algorithms.

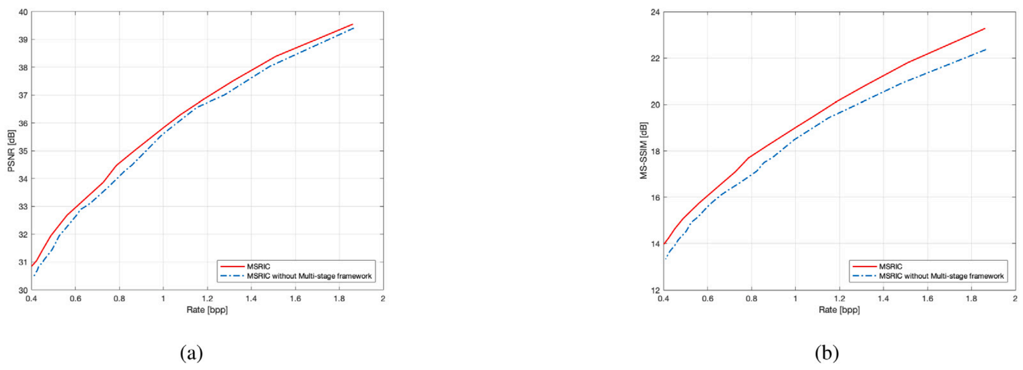

Fig. 11. Ablation results on multi-stage framework. (a) Rate-Distortion(PSNR). (b) Rate-Distortion(MS-SSIM).

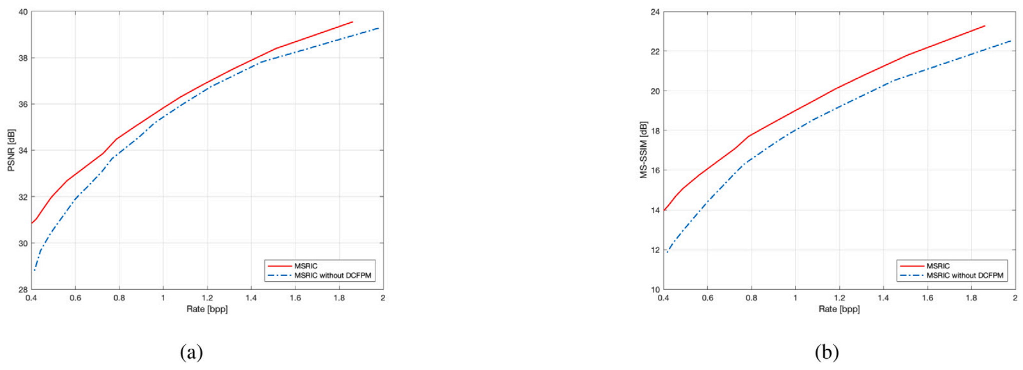

Fig. 12. Ablation results on data-dependent channel-wised probability model. (a) Rate-Distortion(PSNR). (b) Rate-Distortion(MS-SSIM).

In general, our proposed method is able to obtain a better perceptual performance than conventional image coders according to the MS-SSIM performance, which is match with the human visual system. Moreover, despite improvement degrades at lower bit-rates, the overall performance of MSRIC at a large rate scale is better than other algorithms.

To intuitively present the performance improvement of MSRIC, we visualize the reconstructed images of different algorithms in Fig. 10. As shown in the figure, these images are compressed at approximately 0.6 bpp. Our proposed MSRIC shows the best MS-SSIM performance among comparison algorithms, which indicates the best subjective quality. Let us focus on the door in the view, it can be observed that MSRIC reserves the most detail texture information. The latch indicated by blue squares and the pattern indicated by yellow squares are more clear in the image reconstructed by MSRIC. The richer texture information guarantees better perceptual performance, which complies with the results presented by RD curves.

4.4. Complexity performance

The computational complexity and the volume of network parameters are evaluated in this section. We performed our experiments on a computer with a Inter(R) Xeon(R) Processor E5-2630 v4, 256 GB of RAM and a GeForce RTX 2080Ti GPU. The average running time (one complete encoder and decoder process) over Kodak dataset of our proposed MSRIC and other methods were evaluated under exactly

Table 1
Complexity Performance of different compression methods.

Method	Time (s)	Parameters (M)
JPEG	0.15	–
JPEG2000	0.46	–
BPG	0.85	–
Ballé [4]	7.08	6.7
Cheng [16]	16.73	17.8
Cheng [18]	6.64	1.8
MSRIC	7.76	16.2

same environment. For deep learning-based methods, the volumes of network parameters were also calculated. The complexity performance are presented in Table 1.

According to the second column of Table 1, deep learning-based methods generally requires more running time than conventional compression schemes. We can observe that [16] takes the longest time to encode and decode an image. For one thing, this is because it possesses the largest capacity of model parameters. Specifically, the parameters of the additional network for Gaussian mixture models are already 8.2M. For another, there are total two separate entropy coders needed to be implemented to encode the extracted latent representation and the side-information of Gaussian mixture models respectively. Therefore, the total running time of [16] is approximately two times longer than other learning-based approaches, despite of the best RD performance. The complexity may limit its application in real-time scenarios. Compared with [4] and [18], our MSRIC takes more time to encode and decode an image, resulting mainly from the increasing parameter volume. However, we notice that the time-consuming growth are relatively small. From our experiments, the average running time of encoder network and decoder network of MSRIC is only 0.45. Therefore, the above results infer that the running time of deep learning-based image compression might be further reduced by adopting more efficient entropy encoder. In general, MSRIC is an effective image compression algorithm with good perceptual performance.

4.5. Ablation studies

To validate the performance improvement of introducing multi-stage representation and DCFPM, we separately tested the effect of these two components in this section. The training settings were exactly the same as the experiments in Section 4.1 for a fair comparison. Both the PSNR value and MS-SSIM matrix were evaluated to prove the improvement.

To demonstrate the effect of multi-stage representation, we tested the network performance without the multi-stage framework. Specifically, the latent representations extracted by the first three stages were removed while only kept those representations from the final output layer. To ensure the comparison as fair as possible, the number of output channels was set to 64, which was exactly the same as the total number of channels in the multi-stage framework. The comparison result is presented in Fig. 11. According to this figure, introducing the multi-stage representation framework brings improvements on both PSNR and MS-SSIM index at a wide range of bit-rate. Specifically, at the same bit-rate, PSNR value has been improved by about 0.3 dB and MS-SSIM has been improved by about 0.5 dB. It is intuitive that the detail information of texture is gradually lost as the network depth grows. However, a deeper network is more beneficial for extracting structural information and obtain a more compact representation. Both of these two kinds of information are important for image reconstruction. Therefore, the compression efficiency of CAE can be improved, thanks to the multi-stage framework.

For the data-dependent channel-wised factorized probability model, a comparison experiment was carried out by replacing the DCFPM with ordinary channel-wised entropy models as implemented in [32]. For ordinary channel-wised models, it is assumed that all the pixels that belong to the same channel follow one single probability distribution, whatever the input image looks like. That is, there were only 64 sampled piecewise linear functions in total to be optimized during the training process. In other words, the entropy model was independent of batch size. The comparison result is presented in Fig. 12. It can be seen that both PSNR and MS-SSIM performance are improved by introducing data-dependent entropy models during the training process. In general, the accuracy of estimated entropy directly effects the training result. The potential performance of the network can be maximized only when an accurate entropy model is utilized. Moreover, we notice that the performance improvement at lower bit-rate is more distinct. The improvement of PSNR and MS-SSIM even achieve 2 dB at 0.4 bpp. This is because the bit limitation is stricter at lower bit-rate, so that the requirement for accuracy of entropy estimation is stricter. Therefore, an accurate entropy model tends to be more important for image autoencoder at higher compression ratios. To conclude, DCFPM is more accurate than the ordinary channel-wised model, according to the above-mentioned results.

5. Conclusion and future work

In this paper, a deep image compression method with multi-stage representation (MSRIC) was proposed. A multi-stage architecture was specifically designed for image compression task in order to improve compression efficiency. Moreover, considering the importance of an accurate entropy model for the actual performance of a trained network, a data-dependent channel-wised factorized probability model (DCFPM) is proposed for the training process. Then experiments were carried out to evaluate the performance of our proposed method. According to experimental results, our method achieved better MS-SSIM performance compared with existing conventional image compression standards and other deep image compression algorithms that both use factorized entropy models. Ablation studies were presented to demonstrate the validation of the multi-stage framework and the data-dependent probability model.

To further improve MSRIC, future work could investigate more efficient network architecture to reduce parameter volume and/or low-complexity hierarchical hyper-prior for more accurate entropy estimation. More advanced objective functions such as MS-SSIM loss and deep learning-based perceptual loss would be tried to further satisfy human vision system. Moreover, efficient entropy coding method for large context could also be explored to improve both the coding speed and efficiency.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research work was supported in part by the National Science Foundation of China (62071369), and the Key Research and Development Program of Shaanxi Province, China (2020KW-009).

References

[1] G.K. Wallace, The JPEG still picture compression standard, IEEE Trans. Consumer Electron. 38 (1) (1992) 18–34.

[2] A. Skodras, C. Christopoulos, T. Ebrahimi, The jpeg 2000 still image compression standard, IEEE Signal Process. Mag. 18 (5) (2001) 36–58.

[3] G.J. Sullivan, J.-R. Ohm, W.-J. Han, T. Wiegand, Overview of the high efficiency video coding (HEVC) standard, IEEE Trans. Circuits Syst. Video Technol. 22 (12) (2012) 1649–1668.

[4] J. Ballé, V. Laparra, E.P. Simoncelli, End-to-end optimized image compression, in: Proc. Int. Conf. Learn. Represent, 2017.

[5] L. Theis, W. Shi, A. Cunningham, F. Huszár, Lossy image compression with compressive autoencoders, in: Proc. Int. Conf. Learn. Represent, 2017.
[6] F. Mentzer, E. Agustsson, M. Tschannen, R. Timofte, L. Van Gool, Conditional probability models for deep image compression, in: Proc. IEEE Conf. Comput. Vis. Pattern Recog., 2018, pp. 4394–4402.
[7] J. Ballé, D. Minnen, S. Singh, S.J. Hwang, N. Johnston, Variational image compression with a scale hyperprior, in: Proc. Int. Conf. Learn. Represent, 2018.
[8] G. Toderici, S.M. O'Malley, S.J. Hwang, D. Vincent, D. Minnen, S. Baluja, M. Covell, R. Sukthankar, Variable rate image compression with recurrent neural networks, in: Proc. Int. Conf. Learn. Represent, 2016.
[9] G. Toderici, D. Vincent, N. Johnston, S. Jin Hwang, D. Minnen, J. Shor, M. Covell, Full resolution image compression with recurrent neural networks, in: Proc. IEEE Conf. Comput. Vis. Pattern Recog., 2017, pp. 5306–5314.
[10] N. Johnston, D. Vincent, D. Minnen, M. Covell, S. Singh, T. Chinen, S. Jin Hwang, J. Shor, G. Toderici, Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks, in: Proc. IEEE Conf. Comput. Vis. Pattern Recog., 2018, pp. 4385–4393.
[11] E. Agustsson, M. Tschannen, F. Mentzer, R. Timofte, L.V. Gool, Generative adversarial networks for extreme learned image compression, in: Proc. Int. Conf. Comput. Vis., 2019, pp. 221–231.
[12] M. Akbari, J. Liang, J. Han, DSSLIC: Deep semantic segmentation-based layered image compression, in: Proc. IEEE Int. Conf. Acoust. Speech Signal Process., 2019, pp. 2042–2046.
[13] C. Huang, H. Liu, T. Chen, S. Pu, Q. Shen, Z. Ma, Extreme image compression via multiscale autoencoders with generative adversarial optimization, in: Proc. IEEE International Conference on Visual Communications and Image Processing, 2019.
[14] L. Zhou, C. Cai, Y. Gao, S. Su, J. Wu, Variational autoencoder for low bit-rate image compression, in: Proc. IEEE Conf. Comput. Vis. Pattern Recog. Worksh., 2018, pp. 2617–2620.
[15] H. Liu, T. Chen, Q. Shen, Z. Ma, Practical stacked non-local attention modules for image compression, in: Proc. IEEE Conf. Comput. Vis. Pattern Recog. Worksh., 2019, pp. 3–6.
[16] Z. Cheng, H. Sun, M. Takeuchi, J. Katto, Learned image compression with discretized Gaussian mixture likelihoods and attention modules, in: Proc. IEEE Conf. Comput. Vis. Pattern Recog., 2020.
[17] Z. Cheng, H. Sun, M. Takeuchi, J. Katto, Deep convolutional autoencoder-based lossy image compression, in: Proc. Pict. Coding Symp., 2018, pp. 253–257.
[18] Z. Cheng, H. Sun, M. Takeuchi, J. Katto, Energy compaction-based image compression using convolutional autoencoder, IEEE Trans. Multimed. 22 (4) (2020) 860–873.
[19] J. Campos, S. Meierhans, A. Djelouah, C. Schroers, Content adaptive optimization for neural image compression, in: Proc. IEEE Conf. Comput. Vis. Pattern Recog. Worksh., 2019.
[20] M. Li, W. Zuo, S. Gu, J. You, D. Zhang, Learning content-weighted deep image compression, IEEE Trans. Pattern Anal. Mach. Intell. (2020) http://dx.doi.org/10.1109/TPAMI.2020.2983926.
[21] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: Proc. IEEE Conf. Comput. Vis. Pattern Recog., 2009, pp. 248–255.
[22] D. Kingma, J. Ba, Adam: A method for stochastic optimization, Comput. Sci. (2014).
[23] F. Dufaux, G.J. Sullivan, T. Ebrahimi, The JPEG XR image coding standard [Standards in a Nutshell], IEEE Signal Process. Mag. 26 (6) (2009) 195–204.
[24] libjpeg, http://libjpeg.sourceforge.net/.
[25] OpenJPEG, http://www.openjpeg.org.
[26] JPEG XR reference codec, https://jpeg.org/jpegxr/software.html.
[27] libbpg, https://bellard.org/bpg/.
[28] VVC Test model, https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM.
[29] J. Ballé, S.J. Hwang, N. Johnston, M. David, Tensorflow/compression, 2018, https://github.com/tensorflow/compression.
[30] J. Bégaint, F. Racapé, S. Feltman, A. Pushparaja, CompressAI: a PyTorch library and evaluation platform for end-to-end compression research, 2020, arXiv preprint arXiv:2011.03029.
[31] J. Ballé, Efficient nonlinear transforms for lossy image compression, in: Pict. Coding Symp., 2018, pp. 248–252.
[32] T. Dumas, A. Roumy, C. Guillemot, Autoencoder based image compression: can the learning be quantization independent? in: Proc. IEEE Int. Conf. Acoust. Speech Signal Process., 2018, pp. 1188–1192.