IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 21, 2022, accepted May 2, 2022, date of publication May 9, 2022, date of current version May 13, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3173256

Variable-Rate Deep Image Compression With Vision Transformers

BINGLIN LI[1], JIE LIANG[1], (Senior Member, IEEE), AND JINGNING HAN[2], (Senior Member, IEEE)

[1]School of Engineering Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada
[2]WebM Codec Team, Google LLC, Mountain View, CA 94043, USA

Corresponding author: Jie Liang (jiel@sfu.ca)

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada under Grant RGPIN-2020-04525, and in part by the Google Chrome University Research Program.

ABSTRACT Recently, vision transformers have been applied in many computer vision problems due to its long-range learning ability. However, it has not been throughly explored in image compression. We propose a patch-based learned image compression network by incorporating vision transformers. The input image is divided into patches before feeding to the encoder and the patches are reconstructed from the decoder to form a complete image. Different kinds of transformer blocks (TransBlocks) are applied to meet the various requirements in the subnetworks. We also propose a transformer-based context model (TransContext) to facilitate the coding based on previously decoded symbols. Since the computational complexity of the attention mechanism in transformers is a quadratic function of the sequence length, we partition the feature tensor into different segments and conduct the transformer in each segment to save computational cost. To alleviate the compression artifacts, we use overlapping patches and apply an existing deblocking network to further remove the artifacts. At last, the residual coding scheme is adopted to get the compression performance for variable bit rates. We show that our patch-based learned image compression with transformers obtain 0.75dB improvement in PSNR at 0.15bpp than the prior variable-rate compression work on the Kodak dataset. When using the residual coding strategy, our framework keeps good performance in PSNR and is comparable to BPG420. For MS-SSIM, we get higher results than BPG444 across a range of bit rates (0.021 at 0.21bpp) and other variable-rate learned image compression models at low bit rates.

INDEX TERMS Learned image compression, transformer, variable-rate.

I. INTRODUCTION

Recently, there has been a line of researches [1]–[8] on deep image compression. The autoencoder approaches [6]–[8] with the joint autoregressive and hierarchical hyperprior models have been the mainstream practice for learning-based image compression. Although the above methods show promising compression performance compared with conventional image codecs, there are two main drawbacks in real applications.

Firstly, a separate model needs to be trained for each bit rate which increases the coding complexity. To this end, variable bitrate image compression models [9]–[11] are developed to cover various bit rates with one training model. In particular, in [11], a layered coding scheme is developed, where the base layer feature map is obtained by a deep learning (DL) network, and the residual between the input and the base layer reconstruction is coded by a traditional method to cover more bit rates. Motivated by [11], in this paper, we propose a more effective learned image framework by incorporating transformers in the base layer and apply the residual coding to achieve compression across a range of bit rates with one single model. In [11], only eight feature maps are used for the compact representation which limits the learning capability of the base layer. In our framework, a hyperprior network [12] is adopted to estimate the distribution parameters of the quantized feature representation so that the channel dimension of the representation can be set larger in the base layer. Experimental results show 0.75dB improvement in PSNR at 0.15bpp than [11]. When using the residual coding strategy, our framework keeps good performance in PSNR and is comparable to BPG420, whereas the performance in [11] is lower than BPG420.

The associate editor coordinating the review of this manuscript and approving it for publication was Yizhang Jiang.

Secondly, although the masked convolutional context model in [6]–[8] enables to achieve better compression performance compared with the scale-only hyperprior model [12], it brings in extra computational overhead because of the sequential decoding process. Besides, the context information is constrained into 5×5 windows. In this paper, we leverage transformers to capture the long-range dependency and use the masked multi-head attention module in transformers in training to guarantee the causal relationship, which is called TransContext model. Equipped with local transformers, we divide the latent representation into segments. In each segment we conduct the transformer. Each segment can thus be processed parallelly to reduce the total decoding time.

In our framework, only local transformer blocks (TransBlocks) are adopted since the computational complexity of attention mechanism in transformers is a quadratic function to the sequence length. Similar to [13], we divide the input image into image patches. By this way the spatial size of the input can be reduced and we can feed the patch features to transformer blocks in the encoder. At the output of the decoder, an image patch is reconstructed based on the information from the local patch. All the patches are merged into a complete image. This could produce the blocking artifacts at the patch boundaries [14]. To alleviate these artifacts, we use image patches with a small portion of overlaps as the input to the network and average the values of positions where two outputs are predicted. We find this strategy can improve the compression performance to some extent. To further remove the artifacts, a post-processing network [15] is applied.

When the network is designed with fully convolution layers, the size of input image can be arbitrary. Previous works use the entire image as input to the compression network. Patch-based learned image compression has not been explored. Although patch-based image compression methods may result in blocking artifacts as in JPEG [14], it has its advantages. In [16], the inpainting techniques are embedded into the patch-based image compression framework to improve the compression performance. In our work, image patches are used as input to cope with the demanding computation resource for a high-resolution input to transformers.

Our contributions include: 1) We build an effective patch-based learned image compression network with vision transformers in the base layer based on [11] for the variable-rate deep image compression. To alleviate the compression artifacts resulted from patch reconstructions, we partition the image patches with overlaps and utilize an existing deblocking network to further remove the blocking artifacts. 2) Different kinds of transformer blocks are applied to meet the various requirements in the subnetworks. 3) We propose a transformer-based context model to facilitate the Gaussian parameter predictions based on the previously decoded symbols. It is performed on segments of the quantized latent representation and thus can reduce the total decoding time compared with the masked convolution context model in [6].

The rest of the paper is organized as follows. In Section II, we discuss some related work on learned image compression and transformers in vision applications. Then we introduce our framework and explain the building blocks. Experimental results on the Kodak dataset and discussions are presented in Section IV. Section V concludes the paper.

II. RELATED WORKS

A. LEARNED IMAGE COMPRESSION

Many learned image compression models are proposed with the prevalence of DL techniques applied in various research fields. Some researches study learning-based image compression in specific scenarios. In [17], a discrete wavelet transform based DL model is proposed for internet of underwater things. [18] presents a compression model using the convolutional neural network for remote sensing images.

In this paper, we focus on deep image compression for natural RGB images. In [12], a hyperprior network is proposed to learn the scale parameters of the Gaussian scale mixture model for the entropy model. The hyper latent is transmitted as side information to help decode the main latent. However, the estimation is not image-dependent and spatially adaptive after trained. In [6]–[8], the main latent representation is modeled by Gaussian distribution with parameters learned from the context and prior information. The context model allows to combine the information from the neighboring decoded symbols and thus giving a more accurate prediction. It has been the classic learned image compression method due to its superior PSNR and MS-SSIM performance compared with previous works.

In [8], non-local blocks are embedded in the encoder and decoder networks to learn the long-range dependency. However, the self-attention mechanism results in non-negligible computational cost which imposes restrictions to the compression framework design. Based on this joint architecture, recently a new framework that combines the octave convolutions is applied in [19] and achieves higher results than VVC (4:2:0) and other DL-based image compression models. Later works extend the single Gaussian probability model in [6] to Gaussian mixture model (GMM) [20], [21] and show better compression efficiency. In [21], a joint optimization of the image compression and quality enhancement model is applied. The loss at the output of compression network acts as an intermediate supervision and the output of the quality enhancement model is the final reconstruction. The post-processing technique is commonly employed in previous codec JPEG [22] to remove the compression artifacts.

1) VARIABLE-RATE LEARNED IMAGE COMPRESSION

In [23], [24], variable-rate image compression models are proposed based on convolutional and deconvolutional LSTM recurrent networks. In [25], four code layers including a base layer and three enhancement layers are adopted to construct the scalable image compression framework. A decorrelation unit is utilized to obliterate redundancy between the base

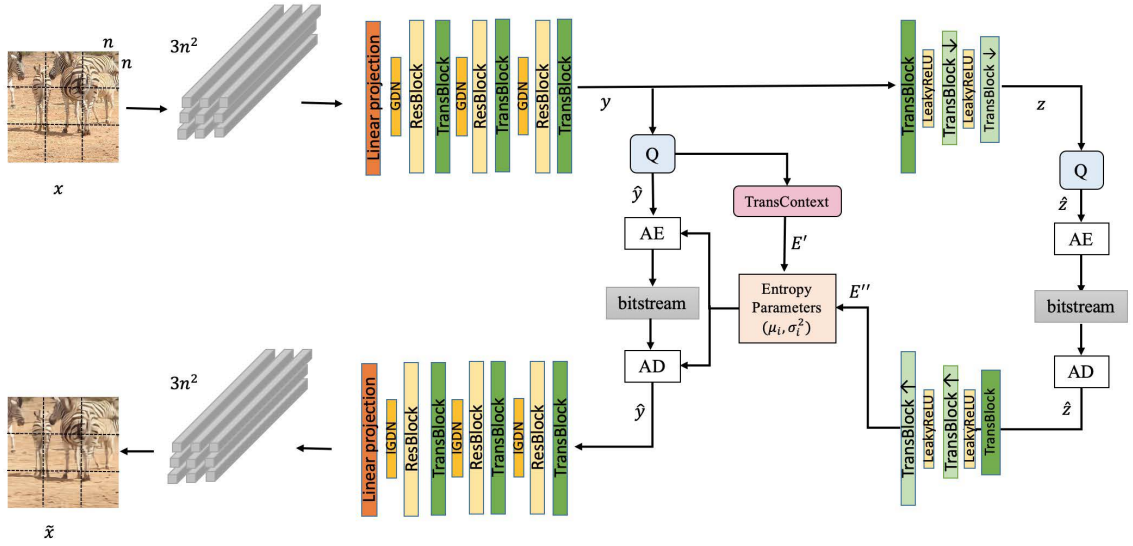


FIGURE 1. Our proposed deep image compression model in the base layer. The input image x is divided into patches with a size of $3 \times n \times n$. Patches are flattened to vectors to formalize a 3D tensor as input to the main encoder. The main decoder outputs a 3D tensor and the vector at each spatial position is reshaped to a $3 \times n \times n$ patch. The patches are merged to the reconstructed image.

layer and the current enhancement layer. During inference, the output of each layer corresponds to the reconstruction at certain bitrate. These methods rely on the layered architectures to adjust for the variable bitrate and are not flexible to obtain a specific rate target. In [9], a multi-scale decomposition transform is learned and a rate allocation algorithm is used to determine the optimal scale of each image block based on content complexity given a target rate. In [10], the authors apply bit-plane decomposition before the transform and introduce a bidirectional network to disentangle the information of different bit-planes. However, the performance of [9] and [10] still has a large gap from the state-of-the-art. In [26], a set of scaling factors is embedded on the quantized feature map from a high bit-rate pre-trained model to fine-tune for the low bit-rate one while keeping main parameters fixed. For low bit rates far from the bit rate of the pre-trained model, the performance is not satisfactory. In [27] a conditional autoencoder is proposed with a coarse rate control by the Lagrange multiplier and a fine-tuning parameter by the quantization bin size. The fine-tuning process is conducted on intervals between individually trained models. Therefore, to get the compression results for a wide range of bitrates, it is still required to train discrete multiple models.

In [11], [28], [29], a hybrid architecture that combines a learning based model and conventional codec is proposed. The BPG-based residual coding is applied as the enhancement layer to obtain compression results for the subsequent bit rates. However, only eight feature maps are used for the compact representation in [11] which limits the learning capability of the base layer. Based on this, we build a more effective model for the base layer.

B. TRANSFORMERS IN VISIONS

Attention mechanisms are widely applied in DL models for speech processing and computer vision problems [20], [30]–[32]. Transformers [33] with multi-head attentions have become predominant DL models for natural language processing (NLP). Due to the ability to learn long-range interactions on sequential data, recently transformers are migrated to many computer vision tasks such as image classification [13], object detection [34], segmentation [35] as well as the low-level computer vision task [36]. However, purely using transformers instead of convolution layers requires to pre-train on a very large-scale dataset and it consumes a vast of time to train [13] to get comparable or even better performance than convolutional networks. Other works integrate convolution layers and transformers to improve results based on similar computation complexity [34], [35], [37].

In [38], transformers are applied on the convolutional feature maps and followed by convolutional decoder to synthesize high-resolution scene images. It leverages the autoregressive structure of transformers to predict current index based on previous indices. This property also suits for the context model in entropy coding module in image compression. In [38], the standard transformer layers are applied. In our work, different transformer modules are developed. In addition, we apply transformers in local windows and symbols in each window can be decoded parallelly, which compensates the expensive time cost from the context model in [6]–[8].

III. OUR APPROACH

We propose an effective learned image framework by incorporating transformers in the base layer and apply the residual coding [11] to achieve compression across a range of bit rates. No previous work on vision transformers is proposed for variable-rate image compression models. Our patch-based framework along with the post-processing step performs better in the base layer than other baselines with residual coding

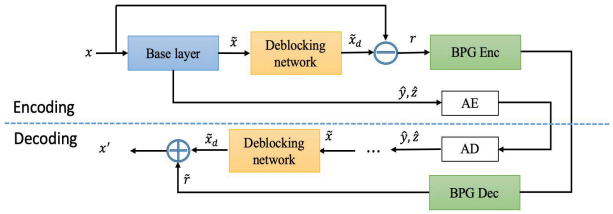


FIGURE 2. The encoding and decoding process of the overall framework. "*Base layer*" is the proposed deep image compression model in Fig. 1. "..." represents the decoder part in the base layer.

scheme [11], [28]. The encoding and decoding process of the overall framework is given in Fig. 2. Next we will elaborate on the autoencoder image compression model in the base layer, deblocking network and residual coding in the enhancement layer separately.

A. AUTOENCODER NETWORK

The architecture of our proposed deep image compression model in the base layer is given in Fig. 1. During training, the input image is randomly cropped with the resolution of 256×256 . Given a 2D image $x \in \mathbb{R}^{H\times W}$, the sequence length is $H \times W$, where H and W are the height and width of the image. As the computational complexity of transformers is a quadratic function of the sequence length, it is infeasible to apply transformers on the entire image directly. We partition the input image x into patches with a size of $n \times n$. Each patch can be flattened to a vector with the length of $3\ n^2$. We have $\frac{H}{n} \times \frac{W}{n}$ patches. Then each vector is projected to d dimension where d is the channel size through the autoencoder network. At this point, we obtain a tensor with $X \in \mathbb{R}^{h\times w\times d}$, where $h = \frac{H}{n}$ and $w = \frac{W}{n}$. We reshape the tensor as $X \in \mathbb{R}^{hw\times d}$ as input of the main encoder network. At the output of the main decoder, the vector at each spatial position is first mapped to $3\ n^2$ dimension from d and then reshaped back to a $3 \times n \times n$ patch. All the patches are merged to a complete image.

The main encoder and decoder consist of Generalized Divisive Normalization (GDN) [39] layers, residual blocks (ResBlocks) [40] and transformer blocks (TransBlocks). GDN layers are suited for Gaussianizing data from natural images. ResBlocks are added to extract local information to compensate the transformer blocks that focus more on long-range dependency. We will introduce the TransBlocks in detail in Sec. III-A1 below.

In Fig.1, we denote the output of the main encoder as y and it is followed by a quantizer Q to obtain the quantized latent $\hat{y}$. Note that $\hat{y}$ has the same spatial size as X with $h \times w$ since no downsampling is needed for the main encoder network. Similar to [12], the latent $\hat{y}$ is modeled with the Gaussian distribution and a hyperprior network is applied to predict the Gaussian parameters μ and σ^2. The hyper-encoder and decoder contain three types of TransBlocks. The output of the hyper-encoder is denoted as z and $\hat{z}$ after quantization. The context model is called TransContext which will be detailed in Sec. III-A2. The output of context model E' is then

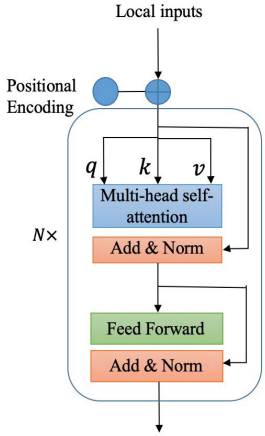


FIGURE 3. The original transformer block in [33].

concatenated with the output from the hyper-decoder E'' to predict the parameters μ and σ^2 for $\hat{y}$. We use the arithmetic encoding AE and arithmetic decoding AD to encode and decode the latent $\hat{y}$ and hyper-latent $\hat{z}$ with predicted Gaussian distribution.

The loss function of the compression model is:

$$\begin{aligned} L_{comp} &= R + \lambda D \\ &= (\mathbb{E}_{x\sim p_x}[-log_2 p_{\hat{y}|\hat{z}}(\hat{y}|\hat{z})] \\ &\quad + \mathbb{E}_{x\sim p_x}[-log_2 p_{\hat{z}}(\hat{z})]) + \lambda D \end{aligned} \tag{1}$$

where the first two items are the bitrate loss for the latent $\hat{y}$ and hyper-latent $\hat{z}$, and the last item D is the distortion function between the original image x and reconstructed image $\tilde{x}$. λ is the tradeoff between the distortion and bitrate. The distortion D can be the mean square error (MSE) loss optimized for peak signal-to-noise ratio (PSNR) or multi-scale structural similarity index measure (MS-SSIM) loss optimized for MS-SSIM [41].

The MSE loss is given below.

$$D_{\text{MSE}} = \frac{1}{N}\sum\sum \|x - \tilde{x}\|^2 \tag{2}$$

where N is the number of elements. The PSNR is calculated by $20\log_{10}\frac{255}{\sqrt{D_{\text{MSE}}}}$. The final compression loss optimized with MSE in our experiment is $L = R + 0.003 \times 255^2 \times D_{\text{MSE}}$ for the base layer.

The PSNR metric is commonly used as the quality assessment for image reconstruction. However, it does not aim for perceived quality. MS-SSIM is a complementary metric to evaluate the structural similarity between two images [41].

The MSS-SSIM is calculated as

$$\text{MS-SSIM}(x,\tilde{x}) = \lfloor l_M(x,\tilde{x})\rfloor^{\alpha_M} \prod_{j=1}^{M} \lfloor c_j(x,\tilde{x})\rfloor^{\beta_j} \lfloor s_j(x,\tilde{x})\rfloor^{\gamma_j} \tag{3}$$

M is the number of scales. $l_M(x,\tilde{x})$, $c_j(x,\tilde{x})$ and $s_j(x,\tilde{x})$ are luminance, contrast and structure comparison measures

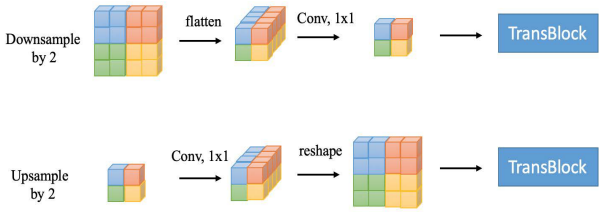


FIGURE 4. DownTransBlock and UpTransBlock illustration.

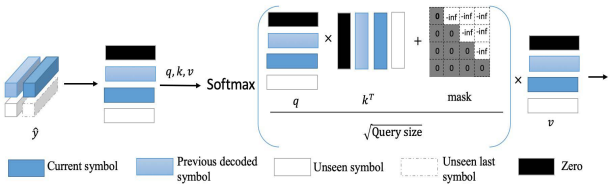


FIGURE 5. Masked attention module in TransContext Model.

respectively [41]. α_M, β_j and γ_j are the relative importance of the terms. The final compression loss optimized with MS-SSIM in our experiment is $L = R + 8 \times (1 - D_{\text{MS-SSIM}})$ for the base layer.

1) TransBlocks

We explore to use transformer blocks to extract the long-range information in the learned image compression network. The original transformer block in [33] is given in Fig. 3. One transformer block contains a multi-head attention network and a point-wise feed-forward network.

We denote the number of heads as m. The input tensor X is divided into m heads with $d_i = \frac{d}{m}$ dimension for each head ($i = 1, 2, \cdots m$). For a tensor $X_i \in \mathbb{R}^{hw \times d_i}$, the multi-head attention process can be represented by a set of equations below. hw is the sequence length and d_i is the vector dimension in ith head. When X is reshaped to a sequence with length of hw, the position information is lost. A positional encoding module is added to provide spatial information at the input.

$$\begin{aligned} Q_i &= X_i W_{Q_i}^T, \quad K_i = X_i W_{K_i}^T, \quad V_i = X_i W_{V_i}^T \\ Z_i &= \text{Softmax}(\frac{Q_i K_i^T}{\sqrt{d_k}}) V_i \\ O(Q, K, V) &= \text{Concat}(Z_1, Z_2, \cdots, Z_m) W_O^T \end{aligned} \tag{4}$$

where W_{Q_i}, W_{K_i}, W_{V_i} and W_O are weights for the linear layers and $\sqrt{d_k}$ is a scaling factor. In the second equation, Softmax is the softmax operation to get the attention scores. In the third equation, the weighted vectors from each head are concatenated as the final output. The attention here is referred as multi-head self-attention (MHSA) mechanism as the three items Q, K and V are obtained from the same input X.

The output of MHSA is then fed into the feed-forward network (FFN) as

$$f(O(Q, K, V)) = ReLU(O(Q, K, V) W_1^T + b_1) W_2^T + b_2 \tag{5}$$

where W_1, W_2 are the weights and b_1, b_2 are the bias of linear layers. RELU is a ReLU activation layer.

Differing from [33] for machine translation tasks, the positional encoding module is based on 2D fixed sine function for images. The periodic property of the sine function allows to extend for longer sequence length. In addition, in order to get a compact feature representation for an image and reconstruct it after the decoder, the transformer blocks need to be scalable for spatial size which is not required in [33] for language modeling. We propose the DownTransBlock and UpTransBlock as depicted in Fig. 4 to meet the various requirements in the architecture.

The regular TransBlock has the input and output with the same spatial size similar to [13]. The DownTransBlock is modified to get the output size reduced by a factor of 2 as shown at first row in Fig. 4. The input tensor is divided into 4×4 blocks. Then we flatten each block to a vector and use a convolution layer with 1×1 kernel size to reduce the channel size to the same as the input tensor. Then the output tensor is followed with the regular TransBlock. The UpTransBlock is the inverse operation of DownTransBlock as shown at the second row in Fig. 4. The DownTransBlock is applied in the hyper-encoder network to obtain the compressed hyper-latent $\hat{z}$ and the UpTransBlock is used to transform back in order to predict the Gaussian parameters for $\hat{y}$.

All the TransBlocks are conducted in local windows. Based on the spatial size of the tensors, we use 8×8 window size for the main encoder and decoder network. Each TransBlock contains $N = 4$ layers of MHSA and FFN. For the hyper-encoder and decoder network, 4×4 window size is applied. Each TransBlock contains $N = 2$ layers of MHSA and FFN.

2) TransContext MODEL

In [6], the context model is a simple masked convolution layer with 5×5 kernels. A symbol is decoded based on previous decoded symbols above and to the left of the current symbol in the window. However, the context information is constrained to local windows. We propose to apply a transformer-based context model which is called TransContext to allow more context to be used for prediction.

During training, we use masked multi-head attention modules [33] in the TransContext model to allow the network to back-propagate for gradient calculation. Fig. 5 gives an illustration of the masked attention module for a tensor with the input size of $2 \times 2 \times d$. The mask shown in the figure has 0s at and below the diagonal direction. The values above the diagonal direction are set to negative infinite.

Given an input from the quantized feature representation $\hat{y}$, we first flatten the tensor and pad it with a vector with all 0s at the beginning. For the current symbol (upper right value of the input), the output of the corresponding position (second vector) only depends on the first vector and padded 0s. In the softmax operation, the product of q and k is added with the mask so that the values corresponding to 0s in the mask will not change and the values above the diagonal direction will

be negative infinite. Note that the softmax of negative infinite is 0. By this way, each symbol only uses the information of previous decoded symbols during test. The output is then sent to the FFN. The last linear layer outputs a tensor with channel size of $2d$ and it is then combined with output from the hyper-decoder network to predict for the μ and σ^2 of Gaussian distribution.

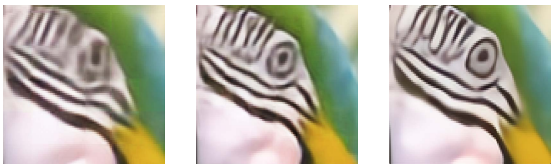
In implementation, given a feature representation $\hat{y} \in \mathbb{R}^{h\times w\times d}$, we divide $\hat{y}$ by 2 in each spatial direction to obtain segments with $\frac{h}{2} \times \frac{w}{2} \times d$ size. In each segment, we apply the TransContext model for inference in parallel. The TransContext model also contains $N = 4$ layers of MHSA and FFN.

B. DEBLOCKING NETWORK

As in Sec. III-A, given an image x, we use image patches as the input in order to leverage the transformer blocks in the autoencoder network. During reconstruction, each vector is reshaped to form an image patch. The patches from all positions are merged to a complete image $\tilde{x}$. Experiments show that the restored image contains some blocking artifacts at the patch borders. This is because each vector only uses the local information in the last linear layer and the edge values for the adjacent patches cannot keep consistency in the prediction. We show two examples in Fig. 6. In (a), the reconstruction is based on non-overlapped image patches, whereas in (b) the image patches are overlapped by two pixels and the overlapping areas are averaged by the neighboring patches. We find that (b) actually has less artifacts than (a).

Although it can reduce some artifacts by using the method in Fig. 6 (b), it is insufficient for image compression where blocking noise can result in PNSR or MS-SSIM degradation. Motivated by [42] that a network is developed to post-process the compression artifacts in JPEG [14] for a better compression performance, we apply the model in [15] to enhance the image reconstruction quality. The decompressed image $\tilde{x}$ is fed into the deblocking network as shown in Fig. 2 to obtain the deblocking image $\tilde{x}_d$. Different from previous work [42] and [31] where only the MSE loss is used during training in accordance with the JPEG optimization metric, we train the deblocking network with MSE or MS-SSIM loss between the deblocking image $\tilde{x}_d$ and the original image x depending on the optimization method of the image compression network. An example result after applying the deblocking network is shown in Fig. 6 (c).

The deblocking process does not increase the bitrate, as when we complete the training process, the reconstructed image from the image compression network can be improved by using one feed-forward step from the deblocking network. It can also be trained jointly with the image compression network end-to-end. However, it will increase the model complexity which makes it hard to train the model on one GPU card. Therefore, in our experiment, we train the two networks separately.



(a) (b) (c)

FIGURE 6. Examples of blocking artifacts for patch-based reconstruction: (a) non-overlap, (b) overlap, (c) overlap+deblock. (please zoom in).

C. RESIDUAL ENCODING FOR VARIABLE RATE

Current learned image compression networks achieve the state-of-the-art compression performance but they need to train a separate model for each bit rate. In variable-rate image compression, a single model is trained to get results for a range of bitrate. The fine-tuning trick may reduce the total training time but can only be applied by a trained model from a high bit rate to a close low bit rate. For low bit rates far from the bit rate of the pre-trained model, the performance drops dramatically [26].

Similar to [11], we use the BPG444 codec [1] to encode and decode the residual between the reconstructed image $\tilde{x}_d$ from the deblocking network in Sec. III-B and the original image x as an enhancement layer as shown in Fig. 2. The bit rate of BPG codec is controlled by a quality parameter q. The total bit rate for our framework is the addition of the bitrate R from the base layer in Eq. 1 and the bitrate R_{bpg} from this enhancement layer controlled by q.

IV. EXPERIMENTS

A. DATASET AND TRAINING DETAILS

1) DATASET

Since for the learned image compression model, the input and the ground truth image are the same, no extra labels are needed for the training. In fact, prior work conduct experiments on different training dataset. We use a subset of 40k images from the COCO-2014 set [43] as the training set and compare the results on the popular Kodak PhotoCD dataset [2] and Berkeley Segmentation Dataset (BSD) 100 test dataset [44].

2) TRAINING SETTING

We randomly crop each image by 256×256 during training. The learning rate is set to 0.00003 for the image compression network. We find that a higher learning rate makes it hard for the training to converge. The training lasts 300 epochs and we reduce the learning rate by 0.1 after 180 epochs. We set the batch size as 20. The learning rate for the deblocking network is set to 0.0001. The training lasts 80 epochs and we reduce the learning rate by 0.5 after 40 and 60 epochs. The batch size is set to 8. We experiment on the Pytorch framework [45] and use one TITAN X GPU for the training with the Adam optimizer.

[1] http://bellard.org/bpg
[2] http://r0k.us/graphics/kodak/

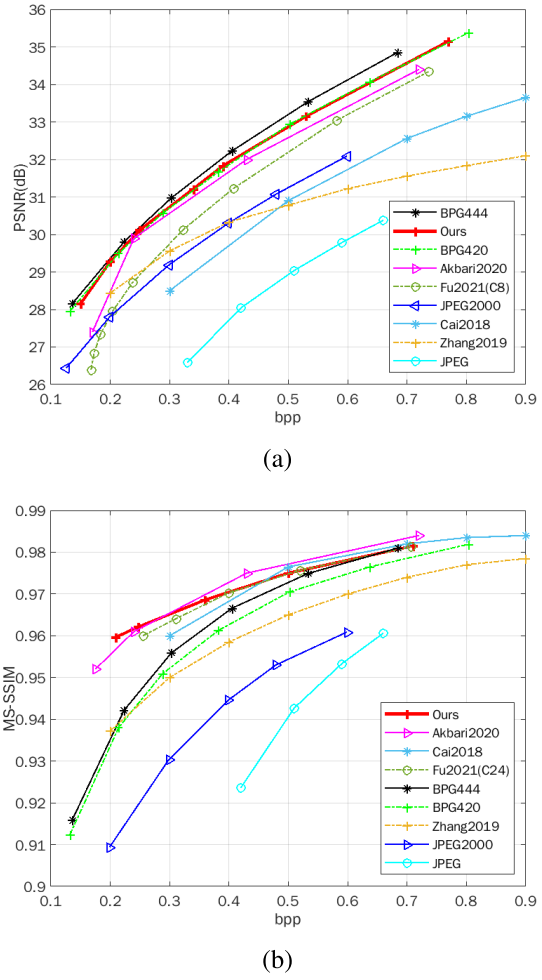


FIGURE 7. (a) PSNR/bpp and (b) MS-SSIM/bpp on the Kodak dataset.

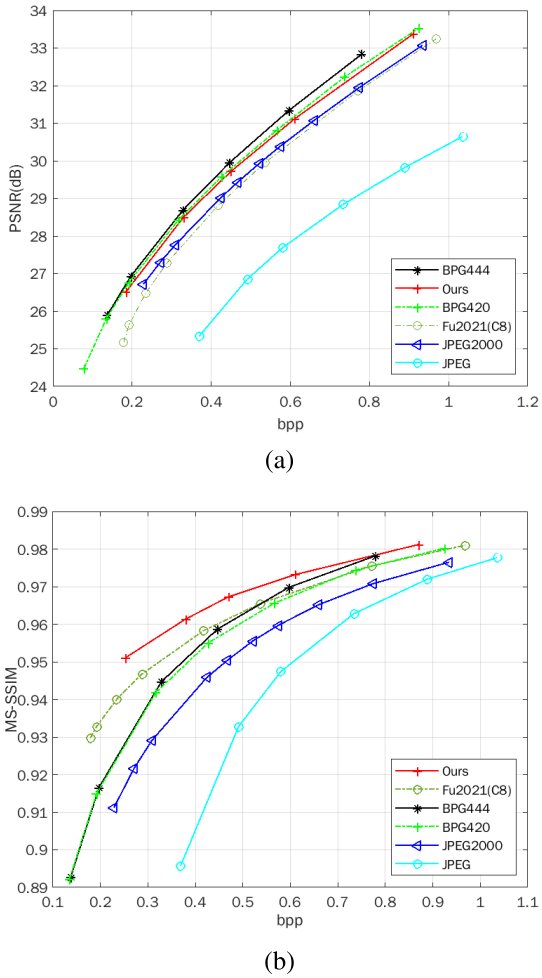


FIGURE 8. (a) PSNR/bpp and (b) MS-SSIM/bpp on the BSD100 dataset.

B. EXPERIMENTAL RESULTS

1) RESULTS ON KODAK DATASET

Fig. 7 shows the comparison of our results with conventional codecs (JPEG [14], JPEG2000 [22], BPG420 and BPG444) and learned variable-rate image compression models *Cai2018* [9], *Zhang2019* [10], *Akbari2020* [11] and *Fu2021* [28] on the Kodak dataset in terms of PSNR and MS-SSIM for per bit per pixel (bpp). Our approach can achieve comparable PSNR with BPG420. The first point in the R-D curve actually reflects the influence of the proposed compression model in the base layer in Fig. 1. Our result in the base layer achieves 0.75dB higher than *Akbari2020* and 1.7dB higher than *Fu2021* at 0.15bpp in which BPG444 is also applied for the residual coding.

For MS-SSIM in (b), we have better performance at the base layer (first point at 0.21bpp) than *Akbari2020* and *Fu2021*. Compared with BPG444, we get 0.021 higher at 0.21 bpp. However, as the bitrate increases, our MS-SSIM result saturates to that from BPG444 which is similar to *Fu2021*. The MS-SSIM of our method shows better performance than the traditional codecs and *Zhang2019*. It also outperforms *Cai2018* at low bit rates. Our approach does not show advantages for MS-SSIM at high bit rates as the residual coding with the classic codec BPG is not optimized for MS-SSIM. However, the residual coding strategy can provide an effective as well as simple way for variable-rate image compression.

2) RESULTS ON BSD100 DATASET

The methods *Cai2018*, *Zhang2019* and *Akbari2020* only show results on Kodak dataset. We also compare our results with JPEG, JPEG2000, BPG420 and BPG444 and the learned variable-rate image compression model *Fu2021* on the BSD100 dataset as given in Fig. 8. The overall trend is consistent to that on Kodak dataset and it shows that the trained models can generalize well.

3) ABLATION TEST: NON OVERLAP vs. OVERLAP

We experiment on two different partition schemes which we call *non-overlap* and *overlap* on Kodak dataset as shown in Fig. 9. For *non-overlap*, we set the patch size with 16×16 . For *overlap*, the patch size is 18 with

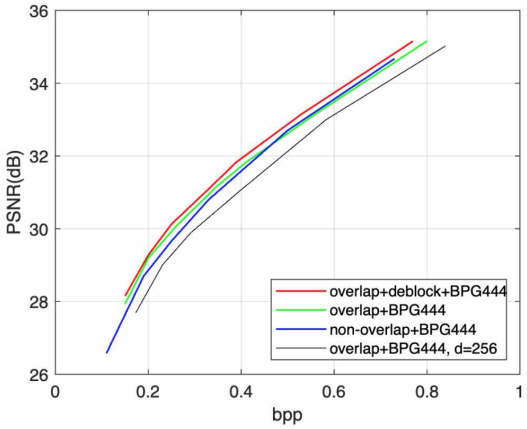


FIGURE 9. Ablation study of our framework.

TABLE 1. Ablation test for TransBlocks and TransContext modules optimized with MSE loss at 0.15bpp.

TransBlocks	TransContext	PSNR	MS-SSIM
√		27.64	0.9324
	√	27.61	0.9251
√	√	27.93	0.9301

TABLE 2. Results on different channel size for d in transformers.

d	bpp	PSNR	MS-SSIM
512	0.15	27.93	0.9301
256	0.17	27.69	0.9286

a stride of 16. The overlapping areas are two pixels in each direction. With the same λ in Eq. 1, the calculated bit rate for *non-overlap* is less than *overlap* at the base layer. After using the BPG residual coding, *overlap* (green curve) show generally better PSNR performance than *non-overlap* (blue curve). For *non-overlap*, only the local information is used to construct each patch in the last linear layer and the edge values for the neighboring patches could have a large variance. For *overlap*, the inconsistency is averaged to reduce the blocking artifacts as shown in Fig. 6 (b).

4) ABLATION TEST: TransBlocks AND TransContext

To prove the effectiveness of the TransBlocks and TransContext, we experiment to delete them respectively and keep the remaining parts of the model same. Tab. 1 shows the results with MSE optimization at 0.15bpp without the deblocking post-processing. The first row shows the result without the TransContext model. The second row gives the result without the TransBlocks in the main encoder and decoder. We show the result for the model with both modules in the third row.

Compared with convolutional layers where the respective field size is constrained by the kernel size, TransBlocks can extract long-range dependency from the feature tensor. When combined with the ResBlocks, our model extracts the local and global information to optimize the compression loss. The TransContext model allows to predict the Gaussian parameters from previously decoded symbols which contributes to a more accurate probability estimation for the arithmetic coding. Tab. 1 shows that both the TransBlocks and TransContext can help improve the compression performance.

5) ABLATION TEST: DEBLOCKING NETWORK

The deblocking network is applied after we get the reconstructed results from the mean decoder. We get about 0.2dB improvement in PSNR and 0.001 in MS-SSIM for *overlap* after the deblocking network. The PSNR curve is displayed with the red curve in Fig. 9.

6) ABLATION TEST: FEATURE SIZE IN TRANSFORMERS

In the above experiment, we set the channel dimension $d = 512$. This can better maintain the information from a patch. We experiment on a smaller channel dimension with $d = 256$. It shows that $d = 512$ can achieve higher PSNR and MS-SSIM at less bit rate as given in Tab. 2. Therefore, we use 512 as the channel dimension for other experiments. The PSNR with $d = 512$ after residual coding (green curve) is steadily better than that with $d = 256$ (black curve) at various bit rates as shown in Fig. 9.

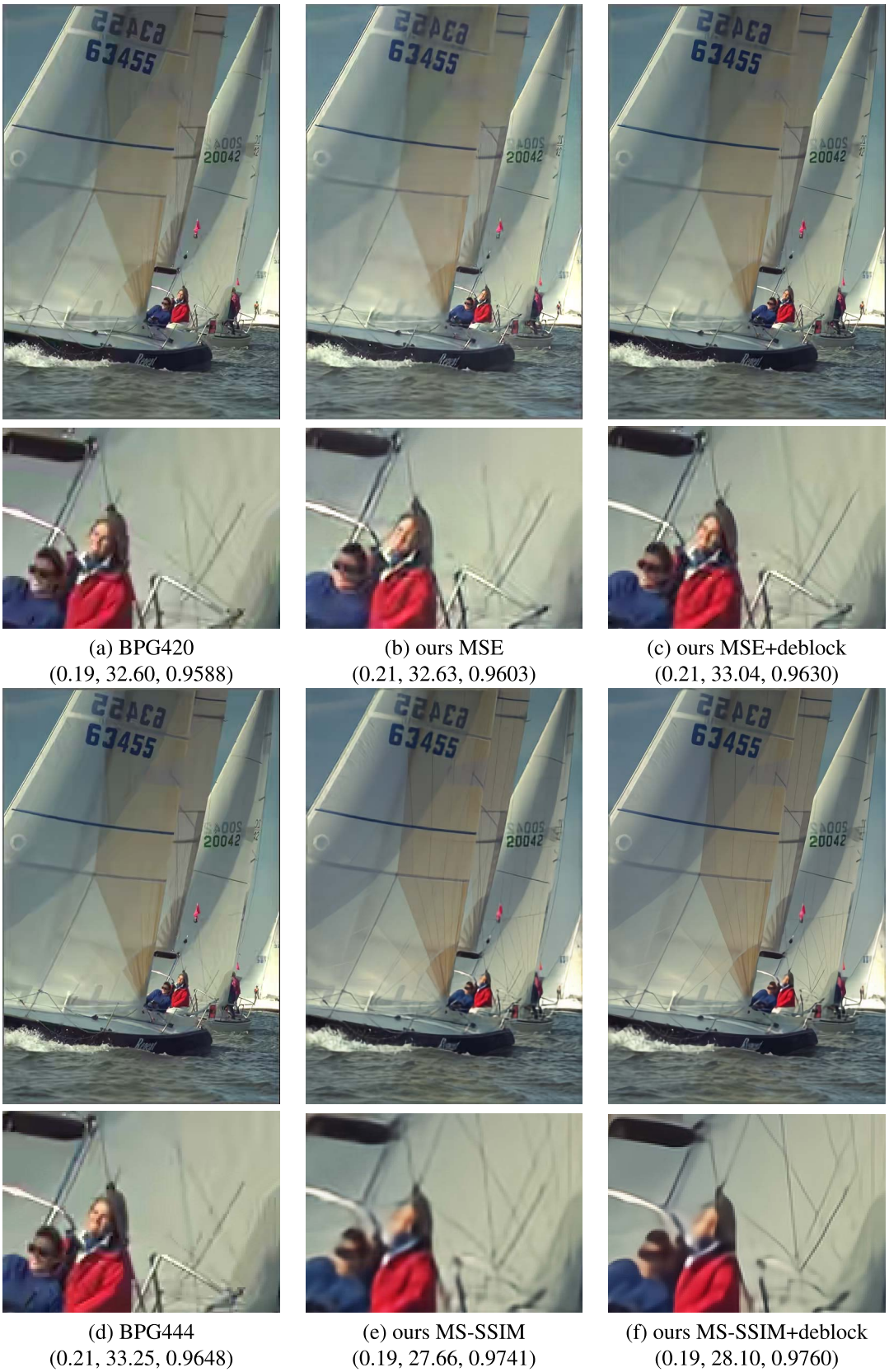
C. TIME COMPLEXITY

We discuss the running time of our framework for inference on a E5-2620 v4 CPU (2.10GHz) with 128GB RAM. The most time consuming part of the model is the context model which needs to be decoded sequentially from previously decoded symbols. For main encoder and decoder networks, it takes about 0.05 s and 0.04 s for one forward step. For the hyper-encoder and hyper-decoder networks, the running time is 0.01 s and 0.01 s. The hyper-latent $\hat{z}$ can be encoded and decoded in parallel. The decoding time for one position is around 0.04s. In [6] the latent $\hat{y}$ can only be decoded one by one and it takes $h \times w$ times of forward steps of the entropy model. In our framework, the transformer is applied on the local windows with size $\frac{h}{2} \times \frac{w}{2}$ and the forward steps is reduced by $\frac{1}{4}$ when decoding parallelly. The running time for one window is around 177s. Note that the arithmetic coding in our experiment is not optimized .[3] Since different platforms may affect the time elapse, we also test the original context model with a masked convolution layer (5×5 kernels) on this device. The running time is about 240s, which takes longer than our scheme.

D. EXAMPLES

In Fig. 10 and Fig. 11, we show reconstructed examples from different methods. In Fig. 10, the results in (e) and (f) show more clear lines on the sail nevertheless blurry human faces.

[3]https://github.com/nayuki/Reference-arithmetic-coding



(a) BPG420
(0.19, 32.60, 0.9588)

(b) ours MSE
(0.21, 32.63, 0.9603)

(c) ours MSE+deblock
(0.21, 33.04, 0.9630)

(d) BPG444
(0.21, 33.25, 0.9648)

(e) ours MS-SSIM
(0.19, 27.66, 0.9741)

(f) ours MS-SSIM+deblock
(0.19, 28.10, 0.9760)

FIGURE 10. Reconstructed example from different methods(bpp, PSNR, MS-SSIM).

(a) BPG420 (0.49, 33.61, 0.9719)
(b) ours MSE (0.45, 32.64, 0.9683)
(c) ours MSE+deblock (0.44, 32.90, 0.9695)
(d) BPG444 (0.51, 33.98, 0.9752)
(e) ours MS-SSIM (0.47, 30.06, 0.9722)
(f) ours MS-SSIM+deblock (0.47, 32.80, 0.9759)

FIGURE 11. Reconstructed example from different methods(bpp, PSNR, MS-SSIM).

The results in (b) and (c) contain more detailed features on human faces. This is because the results in (e) and (f) are obtained from the model in the base layer trained with MS-SSIM loss. As the MS-SSIM loss focuses more on overall structures, the MS-SSIM in (e) and (f) are higher than BPG444, whereas the PSNR in (e) and (f) are relatively low.

At higher bit rate in Fig. 11, the visual difference is not that significant. The MS-SSIM in (f) is slightly better than BPG444 in (d) with less 0.04bpp. Note that in (e) and (f), due to the residual coding scheme based on BPG which is optimized with MSE, the results in (e) and (f) also have high PSNR values. The wall texture on the left in (a) obtained with BPG420 method is not well restored. The corner between the roof and wall contour on the left in (d) is blurry. In both figures, our results are improved when adding the deblocking modules compared with that from the model optimized with the corresponding loss.

V. CONCLUSION

We propose to incorporate vision transformers into a variable-rate learned image compression framework. Different transformer blocks are applied to meet the various requirements in the subnetworks. Compared with other variable-rate learned image compression networks, our framework can get higher PSNR across a range of bit rates and MS-SSIM performance at low bit rates. Ablation experiment shows the effectiveness of the proposed TransBlocks and TransContext model. We also experiment on two different image patch strategies and show that the *overlap* partition achieves better compression performance than the *non-overlap* partition. At last we discuss the time complexity of our model and it can reduce the inference time for the autoregressive context model.

When applying vision transformers, the sequence length which is the number of image patches in our framework associates with the computation cost. More layers of the transformer block can be added and explored if the sequence length can be further reduced. In the future work, we may mask out some of the patches and apply image inpainting techniques to fill the masked patches at the decoder.

REFERENCES

[1] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," 2016, *arXiv:1611.01704*.

[2] E. Agustsson, F. Mentzer, M. Tschannen, L. Cavigelli, R. Timofte, L. Benini, and L. V. Gool, "Soft-to-hard vector quantization for end-to-end learning compressible representations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1141–1151.

[3] M. Li, W. Zuo, S. Gu, D. Zhao, and D. Zhang, "Learning convolutional networks for content-weighted image compression," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3214–3223.

[4] F. Mentzer, E. Agustsson, M. Tschannen, R. Timofte, and L. V. Gool, "Conditional probability models for deep image compression," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4394–4402.

[5] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Energy compaction-based image compression using convolutional AutoEncoder," *IEEE Trans. Multimedia*, vol. 22, no. 4, pp. 860–873, Apr. 2020.

[6] D. Minnen, J. Ballé, and G. D. Toderici, "Joint autoregressive and hierarchical priors for learned image compression," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 10771–10780.

[7] J. Lee, S. Cho, and S.-K. Beack, "Context-adaptive entropy model for end-to-end optimized image compression," in *Proc. 7th Int. Conf. Learn. Represent.*, May 2019, pp. 1–20.

[8] H. Liu, T. Chen, P. Guo, Q. Shen, X. Cao, Y. Wang, and Z. Ma, "Non-local attention optimized deep image compression," 2019, *arXiv:1904.09757*.

[9] C. Cai, L. Chen, X. Zhang, and Z. Gao, "Efficient variable rate image compression with multi-scale decomposition network," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 12, pp. 3687–3700, Dec. 2018.

[10] Z. Zhang, Z. Chen, J. Lin, and W. Li, "Learned scalable image compression with bidirectional context disentanglement network," in *Proc. IEEE Int. Conf. Multimedia Expo. (ICME)*, Jul. 2019, pp. 1438–1443.

[11] M. Akbari, J. Liang, J. Han, and C. Tu, "Learned variable-rate image compression with residual divisive normalization," in *Proc. IEEE Int. Conf. Multimedia Expo. (ICME)*, Jul. 2020, pp. 1–6.

[12] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," 2018, *arXiv:1802.01436*.

[13] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.

[14] G. K. Wallace, "The JPEG still picture compression standard," *Commun. ACM*, vol. 34, no. 4, pp. 30–44, Apr. 1991.

[15] N. Ahn, B. Kang, and K.-A. Sohn, "Fast, accurate, and lightweight super-resolution with cascading residual network," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 252–268.

[16] D. Liu, X. Sun, and F. Wu, "Inpainting with image patches for compression," *J. Vis. Commun. Image Represent.*, vol. 23, pp. 100–113, Jan. 2012.

[17] N. Krishnaraj, M. Elhoseny, M. Thenmozhi, M. M. Selim, and K. Shankar, "Deep learning model for real-time image compression in Internet of Underwater Things (IoUT)," *J. Real-Time Image Process.*, vol. 17, pp. 2097–2111, May 2019.

[18] B. Sujitha, V. S. Parvathy, E. L. Lydia, P. Rani, Z. Polkowski, and K. Shankar, "Optimal deep learning based image compression technique for data transmission on industrial Internet of Things applications," *Trans. Emerg. Telecommun. Technol.*, vol. 32, Apr. 2020, Art. no. e3976.

[19] M. Akbari, J. Liang, J. Han, and C. Tu, "Generalized octave convolutions for learned multi-frequency image compression," 2020, *arXiv:2002.10032*.

[20] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Learned image compression with discretized Gaussian mixture likelihoods and attention modules," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 7939–7948.

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

[22] *Information Technology JPEG 2000 Image Coding System: Core Coding System*, International Organization for Standardization, Geneva, Switzerland, Dec. 2000.

[23] 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," 2015, *arXiv:1511.06085*.

[24] N. Johnston, D. Vincent, D. Minnen, M. Covell, S. Singh, T. Chinen, S. J. Hwang, J. Shor, and G. Toderici, "Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4385–4393.

[25] Z. Guo, Z. Zhang, and Z. Chen, "Deep scalable image compression via hierarchical feature decorrelation," in *Proc. Picture Coding Symp. (PCS)*, Nov. 2019, pp. 1–5.

[26] T. Chen and Z. Ma, "Variable bitrate image compression with quality scaling factors," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 2163–2167.

[27] Y. Choi, M. El-Khamy, and J. Lee, "Variable rate deep image compression with a conditional autoencoder," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3146–3154.

[28] H. Fu, F. Liang, B. Lei, Q. Zhang, J. Liang, C. Tu, and G. Zhang, "An extended context-based entropy hybrid modeling for image compression," *Signal Process., Image Commun.*, vol. 95, Jul. 2021, Art. no. 116244.

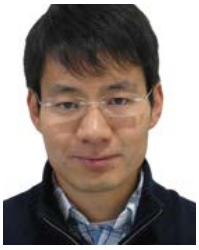
[29] M. Akbari, J. Liang, J. Han, and C. Tu, "Learned multi-resolution variable-rate image compression with octave-based residual blocks," *IEEE Trans. Multimedia*, vol. 23, pp. 3013–3021, 2021.
[30] A. Tursunov, Mustaqeem, J. Y. Choeh, and S. Kwon, "Age and gender recognition using a convolutional neural network with a specially designed multi-attention module through speech spectrograms," *Sensors*, vol. 21, no. 17, p. 5892, Sep. 2021.
[31] B. Li, J. Liang, and Y. Wang, "Compression artifact removal with stacked multi-context channel-wise attention network," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 3601–3605.
[32] K. Muhammad, A. Ullah, A. S. Imran, M. Sajjad, M. S. Kiran, G. Sannino, and V. H. C. de Albuquerque, "Human action recognition using attention based LSTM network with dilated CNN features," *Future Gener. Comput. Syst.*, vol. 125, pp. 820–830, Dec. 2021.
[33] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
[34] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Proc. Eur. Conf. Comput. Vis.*, Cham, Switzerland: Springer, 2020, pp. 213–229.
[35] H. Wang, Y. Zhu, H. Adam, A. Yuille, and L.-C. Chen, "MaX-DeepLab: End-to-end panoptic segmentation with mask transformers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 5463–5474.
[36] H. Chen, Y. Wang, T. Guo, C. Xu, Y. Deng, Z. Liu, S. Ma, C. Xu, C. Xu, and W. Gao, "Pre-trained image processing transformer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 12299–12310.
[37] H. Yan, Z. Li, W. Li, C. Wang, M. Wu, and C. Zhang, "ConTNet: Why not use convolution and transformer at the same time?" 2021, *arXiv:2104.13497*.
[38] P. Esser, R. Rombach, and B. Ommer, "Taming transformers for high-resolution image synthesis," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 12873–12883.
[39] J. Ballé, V. Laparra, and E. P. Simoncelli, "Density modeling of images using a generalized normalization transformation," 2015, *arXiv:1511.06281*.
[40] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[41] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multiscale structural similarity for image quality assessment," in *Proc. 37th Asilomar Conf. Signals, Syst. Comput.*, vol. 2, Jul. 2003, pp. 1398–1402.
[42] C. Dong, Y. Deng, C. C. Loy, and X. Tang, "Compression artifacts reduction by a deep convolutional network," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 576–584.
[43] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, Cham, Switzerland: Springer, 2014, pp. 740–755.
[44] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. 8th IEEE Int. Conf. Comput. Vis. (ICCV)*, vol. 2, Jun. 2001, pp. 416–423.
[45] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, and A. Desmaison, "Pytorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-pe%rformance-deep-learning-library.pdf

BINGLIN LI received the B.Eng. degree in communication engineering from Wuhan University, Wuhan, China, in 2014, and the M.Sc. degree in engineering from the University of Manitoba, Winnipeg, Canada, in 2016. She is currently pursuing the Ph.D. degree with the Multimedia Laboratory, School of Engineering Science, Simon Fraser University, Burnaby, Canada. Since 2018, she has been a Research Assistant with the Multimedia Laboratory, School of Engineering Science, Simon Fraser University. Her research interests include deep-learning-based image compression and computer vision.

JIE LIANG (Senior Member, IEEE) received the B.E. and M.E. degrees from Xi'an Jiaotong University, China, in 1992 and 1995, respectively, the M.E. degree from the National University of Singapore, in 1998, and the Ph.D. degree from Johns Hopkins University, USA, in 2003.

From 2003 to 2004, he worked with the Microsoft Digital Media Division, Video Codec Group. Since May 2004, he has been with the School of Engineering Science, Simon Fraser University, Canada, where he is currently a Professor. His research interests include image and video processing, computer vision, and deep learning.

Prof. Liang received the 2014 IEEE TCSVT Best Associate Editor Award, 2014 SFU Dean of the Graduate Studies Award for Excellence in Leadership, and 2015 Canada NSERC Discovery Accelerator Supplements (DAS) Award. He served as an Associate Editor for several journals, including IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY (TCSVT), and IEEE SIGNAL PROCESSING LETTERS. He has also served on three IEEE Technical Committees.



JINGNING HAN (Senior Member, IEEE) received the B.S. degree in electrical engineering from Tsinghua University, Beijing, China, in 2007, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of California at Santa Barbara, Santa Barbara, CA, USA, in 2008 and 2012, respectively.

He joined the WebM Codec Team, Google, Mountain View, CA, USA, in 2012, where he is the Main Architect of the VP9 and AV1 codecs, and leads the Software Video Codec Team. He has published more than 60 research articles. He holds more than 50 U.S. patents in the field of video coding. His research interests include video coding and computer science architecture.

Dr. Han received the Dissertation Fellowship from the Department of Electrical and Engineering, University of California at Santa Barbara, in 2012. He was a recipient of the Best Student Paper Award at the IEEE International Conference on Multimedia and Expo, in 2012. He also received the IEEE Signal Processing Society Best Young Author Paper Award, in 2015.