Learning Context-Based Nonlocal Entropy Modeling for Image Compression

Mu Li\(^\text{a}\), Kai Zhang\(^\text{b}\), Jinxing Li\(^\text{c}\), Wangmeng Zuo\(^\text{c}\), Senior Member, IEEE, Radu Timofte, Member, IEEE, and David Zhang\(^\text{c}\), Life Fellow, IEEE

Abstract—The entropy of the codes usually serves as the rate loss in the recent learned lossy image compression methods. Precise estimation of the probabilistic distribution of the codes plays a vital role in reducing the entropy and boosting the joint rate-distortion performance. However, existing deep learning based entropy models generally assume the latent codes are statistically independent or depend on some side information or local context, which fails to take the global similarity within the context into account and thus hinders the accurate entropy estimation. To address this issue, we propose a special nonlocal operation for context modeling by employing the global similarity within the context. Specifically, due to the constraint of context, nonlocal operation is intractable in context modeling. We exploit the relationship between the code maps produced by deep neural networks and introduce the proxy similarity functions as a workaround. Then, we combine the local and the global context via a nonlocal attention block and employ it in masked convolutional networks for entropy modeling. Taking the consideration that the width of the transforms is essential in training low distortion models, we finally produce a U-net block in the transforms to increase the width with manageable memory consumption and time complexity. Experiments on Kodak and Tecnick datasets demonstrate the priority of the proposed context-based nonlocal attention block in entropy modeling and the U-net block in low distortion situations. On the whole, our model performs favorably against the existing image compression standards and recent deep image compression models.

Index Terms—Entropy modeling, learned image compression, nonlocal, U-net block.

I. INTRODUCTION

IAGE compression is a crucial problem in computer science that has been studied for decades. Many image compression standards like JPEG [1] have already been used in our daily life. Nevertheless, the last decade has witnessed a population of artificial intelligence and social media, which brings new challenges for sharing and storing huge amount of high-definition media. The requirement for more efficient image compression methods still exists.

Deep neural networks (DNNs) have been proved to be effective and get extraordinary results in many vision tasks such as image restoration [2]–[5], image quality assessment [6], [7], and image classification [8], [9], which throws light on learning better lossy image compression methods. DNNs are natural transforms and fit with the classical transforming coding framework. As a result, most of the existing DNN-based image compression methods follow the transforming coding framework which generally consists of three components, i.e., transform, quantization, and entropy coding. Transform maps the input images to code representations; quantization transforms the representation into discrete counterparts; and entropy coding compresses the quantized codes into the bitstream in a lossless manner. When building the DNN-based transforming coding framework, transforms are replaced with optimizeable DNNs. The entropy of the codes is estimated with DNNs and adopted as the rate loss in a joint rate-distortion optimization.

Entropy modeling that aims to estimate the rate of the codes plays an important role in learned image compression methods. According to Shannon’s source coding theorem [10], given a sequence of codes \( y = \{y_0, \ldots, y_N\} \), the optimal code length of \( y \) should be \( -\sum_{i=0}^{N} \log_2 P(y_i) \), where \( -\sum_{i=0}^{N} \log_2 P(y_i) \) is the entropy. Thus, estimating accurate discrete probability distribution functions (PDFs), i.e., \( P(y_i) \), for the codes is essential in determining the compression rate.

To build accurate and differentiable entropy models, DNNs are adopted to produce parametric PDFs for the codes. Different from codes of the traditional image compression standards,
Fig. 1. Illustration of the relationship between image space and code space produced by DNNs. (a) Image and corresponding code maps. (b) Average PSNR in the code space versus the average PSNR in the image space on Kodak dataset when blurring the images with different blur kernels.

the codes produced by DNNs are in the shape of 3-D cuboid which is made up of several 2-D code maps with the same size. As shown in Fig. 1, there is a close relationship between the source image and the code maps produced by DNNs in both visual and statistic aspects. The code maps produced by DNNs have similar content as the source image and keep a lot of structural information. Considering these special information, two different priors, i.e., the hyper-prior [11] and the auto-regressive prior [12], are introduced to design the DNNs for entropy modeling.

The hyper-prior is some side information $\tilde{z}$ summarized from the codes $y$. The hyper-prior entropy model first extracts the hyper-prior from $y$ with a DNN and then predicts the PDFs of $y$ depending on the hyper-prior with another DNN, i.e., $P(y|\tilde{z})$. The hyper-prior entropy model is exactly another transforming coding framework built on the code space. And the hyper-prior is the code of codes.

The auto-regressive prior is also known as context. For a code $y_i \in y$, its context is defined as $\{y_0, \ldots, y_{i-1}\}$, i.e., all the codes ordered before $y_i$ in $y$. With the context, the PDF of code $y_i$ can be represented as $P(y_i|y_0, \ldots, y_{i-1})$. Different from the hyper-prior, the context is in the same domain as the codes and thus contains more details. In traditional context-based adaptive binary arithmetic coding (CABAC) [13] in H.264/AVC, two nearest codes are adopted as the context which brings a clear performance improvement over previous image compression standards. Recent auto-regressive models based on DNNs, including recurrent neural network (RNN) [14], long short-term memory (LSTM) [15], pixel recurrent neural network (PixelRNN) [16], and pixel convolutional neural network (PixelCNN) [17], can employ much larger context for entropy modeling. In learned image compression, masked convolutional networks [12], [18] and improved masked convolutional networks, i.e., context-based convolutional networks (CCNs) [19], are introduced for modeling the context of the codes.

Existing context models for image compression are based on masked convolutional networks and their variants, which have a limited receptive filed and only consider the local context within the receptive field. Furthermore, the convolution operation only considers the relative positions in the receptive filed in optimization. Global information and content similarity in the context are ignored. To exploit the global information and content similarity among the context, we introduce the nonlocal operation for context modeling by regressing the target code $y_i$ as a weighted sum over its context $\{y_0, \ldots, y_{i-1}\}$. The weights for $y_i$ with $0 \leq i < t$ is calculated with respect to the similarity between $y_i$ and $y_i$. However, the target code, $y_i$, itself is unknown in its context, which makes it unable to directly calculate the weights and conduct the nonlocal operation in context modeling. We call this problem as missing target problem.

Considering the structural information of the codes, we finally propose a proxy similarity metric and introduce a context-based nonlocal operation to tackle the missing target problem. $y$ represents the codes in one code plane. In DNN-based framework, we have $M$ such code planes, i.e., $y_0, \ldots, y_{M-1}$. When predicting the nonlocal estimation of the target code $y_{r,t}$, the previous code planes, i.e., $y_0, \ldots, y_{r-1}$, are in its context and could be used for computation. We note that $\xi_{r,t} = \{y_0, \ldots, y_{r-1}\}$ and $y_{r,t}$ are produced by the same image patch and should have close relationship. Without access to the target code $y_{r,t}$, we finally propose a proxy similarity function between $y_{r,t}$ and $y_{r,t}$ by calculating a weighted sum over the similarities between the proxy code vectors $\xi_{r,t}$ and $\hat{\xi}_{r,t}$. We further introduce indicators for the codes to indicate whether the nonlocal estimation is made on content similar codes. The indicators are then used to produce the attention weights in a nonlocal attention block where our model can adopt the nonlocal estimation when the model finds content similar codes and focus on local estimation results when it fails.

Traditional transforms [20] in image compression are linear and invertible. Distortion only arises from the quantization. As for the recent deep lossy image compression methods, transforms are modeled by learnable nonlinear and powerful DNNs. With the quantization operation and the structure of the network, the learned transforms are generally non-
invertible, which tends to encourage discarding the perceptual unimportant information for better visual quality at low bit rates. However, the performance is usually limited at high bit rates where the distortion is required to be as small as possible. Empirically, the width of the transforms, i.e., the number of feature maps at each layer, is effective in reducing the information losing brought by the transforms. Many deep image compression methods [11], [12], [21] suggest adopting wider transforms at high bit rates. For a deep network, the growing of width will inevitably increase the computational complexity and GPU memory consumption. We suggest adopting an U-net like block in the transforms which could help reduce the time complexity and memory usage in the transforms. With the paired down-sampling, up-sampling operations, and skip connection, the U-net-like architecture is not only able to speed up the transforms but also could combine the information in different scales, and facilitate the information propagation.

Finally, we test the proposed context-based nonlocal entropy modeling and U-net blocks in the learned image compression task. We jointly optimize transforms, i.e., analysis transform and synthesis transform built on U-net blocks, and the context-based nonlocal entropy model with respect to the joint rate-distortion loss in an end-to-end manner. Experiments on the Kodak and Tecnick datasets show that the proposed method can perform favorably against state-of-art lossy image compression standards.

II. RELATED WORK
A. DNN-Based Lossy Image Compression Methods

Recent learned lossy image compression methods based on DNNs are optimized for the rate-distortion tradeoff, where the rate loss is usually modeled as the entropy of the codes. Previous methods [21]–[24] only adopt DNNs for transforms and suppose that all the codes are i.i.d. and follow the same PDF for easy-to-handle entropy models. Without accurate estimation of the entropy, the performance of such methods is limited. Arguably speaking, the code representations generated from the image via a highly nonlinear analysis transform still exhibit strong statistical redundancies [11]. DNNs which estimate the conditional PDFs of the codes based on extra information, i.e., context [12], [18], [19], [25], [26] and hyper-prior [11], [12], are adopted for better entropy modeling. In the following, we will give an overview of some representative DNN-based lossy image compression methods.

Toderici et al. [27] presented an RNN to compress 32 × 32 images in a progressive manner. They later extended the job to full-size images and introduced a BinaryRNN for context-based entropy modeling [25]. Johnston et al. [28] further introduced content-adaptive bit allocation, warm-start training tricks, and perceptual losses to boost the compression performance in terms of multi-scale structural similarity (MS-SSIM).

Ballé et al. [22] started to learn shallow transforms with a GDN activation function in an end-to-end manner and model the entropy with a shared linear piecewise PDF. In their following work [11], each code is supposed to follow a zero-mean Gaussian distribution with the deviation estimated from a side information network depending on the hierarchical hyperprior. Minnen et al. [12] combined the hierarchical hyperprior with context-based auto-regressive prior to boosting the compression performance.

Thesis et al. [23] introduced a straightforward relaxation of the quantization operations and exploited the Gaussian scale mixture for entropy modeling. Rippel and Bourdev [21] proposed a pyramid-based network structure for analysis and synthesis transforms and an adaptive code length regularization for real-time image compression. And a generative adversarial loss [29] was introduced to generate visually better decoded images at low bit rates. Agustsson et al. [24] presented a soft-to-hard quantization scheme with a parametric softmax function.

Li et al. [26] learned an importance map as the side information for content variant rate controlling and exploited a simple convolutional network for context-based entropy modeling. Mentzer et al. [18] proposed a mask convolutional network for context-based entropy modeling and alternatively optimized the entropy model, transforms, and the importance map. Limited by the raster coding order, the context-based models should be decoded in serial coding order and thus are computational inefficiency. Li et al. [19] further introduced a special coding order and group context depending on a code dividing scheme for parallel computation of each group.

Previous deep context-based entropy models only focus on local context in the receptive field of convolutional networks and usually ignore the nonlocal similarity in the global context. In this article, we take the nonlocal similarity within the global context into account and combine it with the local representation for more accurate entropy modeling.

B. Nonlocal Methods for Image Processing

Buades et al. [30] explored the self-similarity among pixels and proposed the nonlocal means based on a content-weighted nonlocal averaging of all pixels for image denoising. Wang et al. [31] formulated the nonlocal operation as a uniform block and adopted it in DNNs to combine local and nonlocal information for object detection. Liu et al. [32] proposed a nonlocal recurrent network which incorporates nonlocal operations into a recurrent network for image restoration. Zhang et al. [33] exploited the nonlocal operation to build attention masks to capture long-range dependency between pixels and pay more attention to the challenging parts for high-quality image restoration. Zhuang et al. [34] employed the global and nonlocal similarity via a low-rank tensor factorization for image denoising. Zhang et al. [35] introduced a nonlocal patch tensor-based visual data completion algorithm with the low-rank constraint for image inpainting.

For learned image compression, Cheng et al. [36] introduced a nonlocal attention block in the encoder–decoder network structure and simplify the block to boost the running speed. Chen et al. [37] also brought the nonlocal network operation in encoder–decoder networks for both the main codes and the hyper-prior codes and an improved context modeling. However, existing jobs only consider the nonlocal operation in producing the codes and images and ignore its effect in context modeling. In this article, we dig out the
nonlocal similarity among the context of codes for more accurate entropy modeling. Specially, with the constraint of context, the target code is unavailable for calculating the nonlocal weights, which makes the original nonlocal operation not applicable on context. In this article, we employ the structural similarity among the 3-D code block and propose a novel proxy nonlocal operation for context modeling. In addition, a confidence metric that measures whether the nonlocal estimation is made on the content similar codes is used to produce the attention weight in the nonlocal attention block. As a result, our nonlocal attention block can adopt the nonlocal estimation when the model finds the content similar codes and adopt local estimation results when it fails.

III. CONTEXT-BASED ENTROPY MODELING

Different from the codes produced in traditional image compression methods, the codes produced by DNNs still keep close relationship with the input image. To support this, we visualize the image and its corresponding code planes in Fig. 1(a). As one can see, the code maps produced by the DNNs have similar content as the original image and keep a lot of structural information. Furthermore, we also conduct a quantitative experiment. We produce similar image pairs by blurring the images with Gaussian blur kernels and separately compute the peak signal-to-noise ratio (PSNR) of these pairs in code space and image space. Ten blur kernels with the size of $9 \times 9$, zero mean, and delta in the range $[0.6, 2.6]$ are adopted in the experiment. Fig. 1(b) shows the relationship between the code space and the image space when blurring the images on Kodak dataset. In general, there is a strong positive correlation between the code maps and the image, which indicates that similar image will produce similar code maps.

Considering the relationship between the codes and the images, many notable features and patterns can be adopt in predict the PDFs of the codes. Context of the codes is subset of the original codes and shares the same distribution with the codes. It contains more detail information surrounding the codes and should help dig out more effective patterns in entropy modeling and further improve the prediction accuracy of PDFs.

Modeling the entropy of the code based on its context is an auto-regression problem where the PDF of a code is regressed from its context. Let $y \in \mathbb{R}^{M \times H \times W}$ denote a 3-D code cuboid generated by the analysis transform, where $M$, $H$, and $W$ separately are the channel, height, and width of the code cuboid. $y_r(p, q)$ represents a code at the position $(p, q)$ in $r$th channel. With a given coding order, the context of a target $y_r(p, q)$ among $y$ is defined as all the codes scanned before it, i.e., $\text{CTX}(y_r(p, q), y)$. Fig. 2(a) gives an example of the context of the target code in red with a raster scanning order.

Convolutional neural networks are efficient in processing 2-D and 3-D media data. Previous context-based entropy modeling usually adopts masked convolutional neural networks to predict the PDFs of the codes. As shown in Fig. 2(b), limited by the structure, convolutional neural networks only focus on the local context surrounding the target code in the receptive field. In addition, the convolution operation only considers the relative positions in the receptive filed in optimization. Thus, the weights are optimized according the relative positions instead of the content similarity between the context and the target code. Global information and content similarity in the context are ignored. To exploit the global information and the content similarity, we introduce the nonlocal operation for context-based entropy modeling.

Nonlocal operation is usually defined as a weighted sum over the whole image, where the distance between a target pixel and other pixels is employed to produce the weights. Similar pixels play a more important role in estimating the target. As for the code block, we calculate the nonlocal representation of a target code $y_r(p, q)$ by summarizing all the codes in $r$th channel of the context with the content-related weights. However, it is unable to calculate the weights when the target code itself is unavailable for computation in the context. Given $\xi_r(p, q) = \{y_0(p, q), \ldots, y_{r-1}(p, q)\}$, the code vector $\xi_r(p, q)$ and the code $y_r(p, q)$ are produced by the same image patch, i.e., $x(p, q)$. Considering the close relationship between the code space and image space, similar codes, i.e., $y_r(p, q)$ and $y_r(u, v)$, lead to similar image patches, i.e., $x(p, q)$ and $x(u, v)$. And similar image patches $x(p, q)$ and $x(u, v)$ in turn lead to similar code vectors, $\xi_r(p, q)$ and $\xi_r(u, v)$. Thus, it is reasonable to evaluate the similarity between $y_r(p, q)$ and $y_r(u, v)$ with the distance between the two proxy vectors, $\xi_r(p, q)$ and $\xi_r(u, v)$. As shown in Fig. 2(c), by finding similar code vectors, we predict the nonlocal representation of the target code in red with the similar codes in blue. Image restoration methods like BM3D [38] argue that there are many similar patches in the natural image. Thanks to the close relationship between the
code space and the image space, there should be many similar codes and code vectors in the code cuboid. Consequently, the introduced nonlocal operation can find these similar codes in the context and employ them for more accurate entropy modeling.

In this article, we introduce a nonlocal attention block that combines the information from the local and global context for entropy modeling. Context-based convolutional networks (CCNs) [19] are adopted to modeling the local context for their efficiency and effectiveness. In the following, we will first give a short description of the CCNs and then introduce the context-based nonlocal operation.

A. CCNs in Modeling Local Context

Given the code block \( y \in \mathbb{R}^{M \times H \times W} \), the output of the \( r \)th CCN layer is a 4-D tensor \( \mathbf{v}^{(r)} \in \mathbb{R}^{M \times H \times W \times N_r} \) with \( N_r \) feature blocks. Each feature block has the same size as the code block. The feature \( v_{i,j}^{(r)}(p,q) \) in r\( h \) channel and \( i \)th feature block at spatial location \((p,q)\) is a representation of \( y_i(p,q) \) and only convey information from the CTX \( y_i(q,p), y \) .

To speed up the efficiency and break up the serial decoding process, the CCNs divide the codes into \( K = M + H + W - 2 \) non-overlap groups and parallel process each group, where \( \text{GP}_k(y) = \{y_i(p,q) | r + p + q = k\} \) denotes the \( k \)th group. Then, the group context is defined as \( \text{PTX}(y_i(p,q), y) = \{y_i(p', q') \mid r' + p' + q' < k\} \). With the group context, codes within each group share the same context and thus can be processed in parallel. Without a clear performance drop in the entropy modeling, the special group context could dramatically accelerate the decoding efficiency.

The CCNs are built on mask convolution layers which are defined as

\[
v_{i,j}^{(r)}(p,q) = \sum_{j=1}^{N_r} \sum_{i=1}^{M} (\mathbf{u}^{(r)}_{i,j} \ast (m_{s,t}^{(r)} \odot w_{i,j,r,s}^{(r)}))(p,q) + b_i^{(r)} \tag{1}
\]

where \( \{i,j\} \) and \( \{r,s\} \) are indexes for the feature block and channel dimensions, respectively. \( \mathbf{u}^{(r)} \) and \( \mathbf{v}^{(r)} \) are the input and output of the \( r \)th convolution layer. \( \mathbf{v}^{(r-1)} \) is activated by a nonlinear element-wise function to produce \( \mathbf{u}^{(r)} \). \( w_{i,j,r,s}^{(r)} \) the weight to connect feature maps \( u_{i,j,r,s}^{(r)}(u,v) \) is the bias term. Given \( w_{i,j,r,s}^{(r)} = \{w_{i,j,r,s}^{(r)}(u,v) \mid -k_s \leq u, v \leq k_s\} \) where \( k_s \) is kernel size, the corresponding mask for the input layer is defined as

\[
m_{s,t}^{(0)}(u,v) = \begin{cases} 1, & \text{if } s + u + v < r \\ 0, & \text{otherwise} \end{cases} \tag{2}
\]

For the \( r \)th hidden layer, the mask is modified to include the codes in the same group

\[
m_{s,t}^{(r)}(u,v) = \begin{cases} 1, & \text{if } s + u + v \leq r \\ 0, & \text{otherwise} \end{cases} \tag{3}
\]

B. Context-Based Nonlocal Operation

Different channels in the code block are generated by different convolutional filters. Only the codes in the same channel are produced by the same transform and thus should follow the same distribution. We only consider the codes in the same channel in the nonlocal operation. Besides, the context should be considered. For the target code \( y_i(p,q) \), masks \( m_i^j \) are introduced to exclude the codes outside of the context. Following the group context PTX \( y_i(p,q), y \) in CCNs, we set the mask \( m_i^j(p,q,u,v) = 1 \) if \( y_i(u,v) \in \text{PTX}(y_i(p,q), y) \), i.e., \( u + v < p + q \). Otherwise, \( m_i^j(p,q,u,v) = 0 \). Fig. 3 shows the codes used in the nonlocal operation with a yellow plane where the masks are 1s.

Evaluating the similarity between the target and the others is necessary for nonlocal operation. Due to the target code is not in the context, it is unable to calculate the distance between the target and others directly. Instead, we propose a proxy similarity function by considering the available codes in the same position but the other channels. The similarity metric \( g_d(y_i(p,q), y, u,v) \) between the code \( y_i(p,q) \) and \( y, u,v \) is defined as

\[
g_d(y_i(p,q), y, u,v) = \sum_{j=0}^{r-1} w_{r,j}^{d}(y_j(p,q) - y_j(u,v)) + b_r. \tag{4}
\]

Here, \( w_{r,j}^{d} \) is the weight to balance the contribution of the proxy codes in different channels. \( w_{r,j}^{d} \) is set to be \( 1/(r + 1) \) in initialization and dynamically optimized with the whole framework. \( b_r \) is the bias term for the proxy similarity metric in the \( r \)th code plane. \( g_d \) is a proxy based on the available proxy codes in the context. When considering more proxy codes in different code planes, the result will be more accurate and close to the real similarity.

The context-based nonlocal operation is defined as

\[
g_{nlc}(y_i(p,q)) = \sum_{u,v} w_{r}^{c}(p,q,u,v)y_i(u,v) \tag{5}
\]

where the weight \( w_{r}^{c}(p,q,u,v) \) is defined as

\[
w_{r}^{c}(p,q,u,v) = \frac{m_i(p,q,u,v)e^{-g_d(y_i(p,q), y, (u,v))}}{\sum_{u',v'} m_i(p,q,u',v')e^{-g_d(y_i(p,q), y, (u',v'))}}. \tag{6}
\]
Let et al.: LEARNING CONTEXT-BASED NONLOCAL ENTROPY MODELING 1137

Fig. 4. Architecture of the proposed lossy image compression method, including an analysis transform $g_a$, an adaptive trainable quantizer $g_q$, a context-based nonlocal entropy model $g_e$, and a synthesis transform $g_s$. Conv: regular convolution with filter support (kernel_size $\times$ kernel_size) and number of channels (output $\times$ input). UnetBlock: $a_0$, $a_1$ and $a_2$ are the down-sampling multipliers in the U-net structure. MConv: mask convolution used in CCNs with filter size (kernel_size $\times$ kernel_size) and number of feature blocks (output $\times$ input). Note that the number of channels is fixed in MConv, and is the same as the input of the entropy model, i.e., $y$. (a) Structure of UnetBlock. (b) Structure of the whole framework.

Compared to the original nonlocal operation, our estimation is made on proxy similarity and could be inaccurate especially in the first several code planes where there are not enough proxy codes for computation. As a result, the nonlocal estimation could be made by those dissimilar codes, resulting in decreased accuracy of the nonlocal estimation. Inaccurate estimation would further lead to poor estimation on the entropy. To indicate whether the target code is estimated by content similar codes and reduce the contribution of nonlocal representation in those bad cases, a confidence indicator is introduced as the weighed similarity among the proxy code vectors. If the nonlocal operation fails to find the content matching codes, the confidence indicator should be very large, and vice versa

$$g_e(y, (p, q)) = \sum_{u, v} w_r^e(p, q, u, v) g_d(y, (p, q), y, (u, v)). \quad (7)$$

We combine the confidence indicators and local representations to generate the attention weights. Then, the Hadamard product of the attention weights and nonlocal representations is then concatenated with the local representations to produce output of the attention block. Fig. 4(b) gives the structure of the block. With the introduced indicator, our attention block can adopt the nonlocal estimation produced by content similar codes and tend to focus on the local information when the nonlocal estimation is made on dissimilar codes.

IV. CONTEXT-BASED NONLOCAL ENTROPY MODELING FOR LOSSY IMAGE COMPRESSION

A lossy image compression method usually consists of transforms based on DNNs and is optimized for a joint rate-distortion objective function. Fig. 4 gives the framework of our lossy image compression method including the analysis transform $g_a$, quantizer $g_q$, synthesis transform $g_s$, and the context-based nonlocal entropy model $g_e$. The analysis transform maps a color image $x$ to the code representation $z$, which is further discretized by $g_q$ to produce the code block $y$. The synthesis transform $g_s$ takes $y$ as input to produce a color image $\hat{x}$ as the reconstruction of $x$. In this section, we will first describe the transforms, quantization function and the objective function and then introduce a post entropy model to simplify the entropy coding.

A. Network Structure for Transforms

DNNs are non-invertible. Transformations based on DNNs will lose some information in mapping the input image to the code representation. Such information loss could be ignored for models at low bit rates when the distortion between the input and the decoded image is large enough. However, with the decrease of the distortion, the information loss brought by DNNs will start to hinder the performance. To reduce the information loss, some learned image compression methods suggest
adopting a wider network at high bit rates. Consequently, the time complexity and memory consumption also increase rapidly. A network structure taking both the computational efficiency and the width into account is needed.

U-net [39], [40] is a light structure proposed for medical image segmentation. With paired down-sampling, up-sampling, operations and skip connections, U-net is fast and can utilize representations in different scales. Besides, the skip connection could facilitate information propagation and ease the training of the network [41]. As shown in Fig. 4(a), we adopt the U-net structure in building the framework [23], an identify mapping \( \hat{x} \) to the nearest quantization center in \( \mathbb{R}^N \) is modeled as a normal convolution to increase the channels of \( z \) by minimizing the mean-squared quantization error

\[
\mathcal{L}_q(\sigma) = \frac{1}{\text{MHW}} \sum_{r,p,q} \| y_r(p,q) - z_r(p,q) \|_2^2. 
\]

We initialize \( \sigma \) as a uniform quantization in (0, 1), which are then optimized and updated according to the distribution of \( z \).

C. Modeling the Objective Function

The whole framework is optimized for a joint rate-distortion objective function, where the distortion loss is directly modeled as the difference between the decoded image and the input image \( x \). Two separate metrics, i.e., standard mean square error (MSE) and the perceptual metric MS-SSIM [46], are adopted as the distortion loss. The MSE distortion loss \( \mathcal{L}_{D_{\text{MSE}}} \) and MS-SSIM distortion loss \( \mathcal{L}_{D_{\text{MS-SSIM}}} \) as follows:

\[
\mathcal{L}_{D_{\text{MSE}}}(x; \phi, \psi) = \frac{1}{3H_1W_1} \| \hat{x} - x \|_2^2 
\]

and

\[
\mathcal{L}_{D_{\text{MS-SSIM}}}(x; \phi, \psi) = 100 - 100 \text{ MS-SSIM}(\hat{x}, x) 
\]

where \( \hat{x} = g_q(z_r(x; \phi; \sigma); \psi) \). \( H_1 \) and \( W_1 \) are separately the height and width of the image \( x \). We denote our method optimized for \( \mathcal{L}_{D_{\text{MSE}}} \) as Ours(MSE) and \( \mathcal{L}_{D_{\text{MS-SSIM}}} \) as Ours(MS-SSIM).

Each code in \( y \) is assumed to follow an MoG distribution depending on its context. We introduce a context-based non-local entropy model \( g_e \) to produce the mixture weight, mean, and variance of these MoG distributions. As shown in Fig. 4, the entropy model consists of a nonlocal attention block, several CCN residual blocks, and three final CCN layers to produce the mean, variance, and weight estimation separately. \( \theta \) is the parameters for the proposed entropy model.

Given \( y_r(p,q) = \omega_{r,j} \) is the \( j \)th quantization center for \( r \) channel, the discretized probability of a code \( y_r(p,q) \) is defined as

\[
P(y_r(p,q); \theta) = \int_{\mathbb{R}^{C-1}} \int_{\mathbb{R}^{N}} \pi_i N(\mu_i, \sigma_i^2) d\alpha \]

where \( C \) is the number of Gaussian distribution in the mixture; \( \pi_i, \mu_i, \) and \( \sigma_i^2 \) are the mixture weight, mean, and variance of the \( i \)th component, respectively. Specially, \( \omega_{r,-1} = -\infty \) and \( \omega_{r,L} = \infty \) for the boundary cases. Then, the entropy of the codes are adopted as the rate loss

\[
\mathcal{L}_R(x; \phi, \theta) = - \sum_{y \in g_e(z_r(x; \phi; \sigma))} \log P(y; \theta). 
\]

Finally, we propose a rate-distortion objective function for the parameters \( \phi, \psi, \theta \) over the training set \( \mathcal{X} \) as

\[
\mathcal{L}(\phi, \psi, \theta) = \sum_{x \in \mathcal{X}} \mathcal{L}_{D}(x; \phi, \psi) + \lambda \mathcal{L}_{R}(x; \phi, \theta) 
\]

where \( \lambda \) is the tradeoff parameter for balancing the rate and distortion in the objective function.
D. Post-processing for Entropy Coding

In entropy coding, a discrete probability table that contains the probability \( P(y, p, q) = \omega_{r,i} \) for \( l = 0, \ldots, L - 1 \) is needed. With the current entropy model built on MoG distributions, we should first produce the parameters for MoG is needed. With the current entropy model built on MoG distributions, we should first produce the parameters for MoG with the trained analysis transform \( g_a \) and then optimize the post entropy model on the extracted code blocks. Finally, we implement our own arithmetic coding with the context-based nonlocal entropy model to compress the extracted code blocks. For the network structure, we only make a small modification on the last layer. Instead of taking three separate CCN layers to produce the mean, variance, and weight estimations used in MoGs, we make use of a single CCN layer followed by a softmax function in (15) with an Adam solver [48]. Starting from a learning rate of 10\(^{-5}\), smaller learning rates, i.e., 10\(^{-6}\) and 10\(^{-7}\), are adopted until the loss does not decrease for 5 successive epochs. The post entropy model is also optimized with the Adam solver in the same way. We train 14 models for seven different bit rates and two distinct distortion metrics, i.e., MSE and MS-SSSIM. For testing, the compression rate is evaluated by bits per pixel (bpp), which is the total amount of bits used to compress the image divided by the whole number of pixels in the image. Two quantitative metrics, i.e., Multi-Scale Structural Similarity (MS-SSSIM) and the PSNR, are considered in evaluating the image distortion.

V. EXPERIMENTS

In this section, we test the proposed context-based nonlocal entropy model and Unet-block in the lossy image compression. Ten thousand high-quality images are collected from the photo-sharing website Flickr and down-sampled to further reduce possible compression artifacts. We crop 640,000 color patches of size 3 × 256 × 256 as the training sets. For the post entropy model, we first extract the code blocks with the analysis transform from the full-size images and then crop code blocks of size \( M \times 60 \times 60 \) for training. We test our models on two benchmark datasets – Kodak and Tecnick [47], and then compare them to the recent deep image compression algorithms and state-of-the-art image compression standards. The pretrained models for testing are available at https://github.com/limuhit/Nonlocal-CCN.

A. Experimental Setup

The quantization centers \( (L) \) is set to be 8 and the number of Gaussian components in MoG \( (C) \) is set to be 3. We follow the warmup strategy in [19] and jointly optimize the transforms and entropy model by minimizing the rate-distortion objective function in (15) with an Adam solver [48]. Starting from a learning rate of 10\(^{-5}\), smaller learning rates, i.e., 10\(^{-6}\) and 10\(^{-7}\), are adopted until the loss does not decrease for 5 successive epochs. The post entropy model is also optimized with the Adam solver in the same way. We train 14 models available at https://github.com/limuhit/Nonlocal-CCN.

B. Quantitative Evaluation

Using MS-SSSIM and PSNR as distortion metrics, we compare our methods with existing image compression standards such as JPEG [1], JPEG2000 [49] and BPG [50] and recent DNN-based compression models in terms of rate-distortion curves. DNN-based compression models include Agustsson17 [24], Theis17 [23], Toderici17 [25], Rippel17 [21], Mentzer18 [18], Johnston17 [28], Li18 [26], Li20 [19], and Minnen et al. [12] (Minnen18). Both JPEG (with 4:2:0 chroma subsampling) and JPEG2000 are based on the optimized implementations in MATLAB2017. For BPG, we adopt the latest version from its official website with the default setting. We find the results of most of the competing methods from https://github.com/tensorflow/compression. For those results that are not reported in the site, we reproduce the results with the given test codes.

Fig. 5 shows the rate-distortion curves on the Kodak dataset. (a) PSNR. (b) MS-SSSIM.

Fig. 6. Rate-distortion curves of different compression methods on the Tecnick dataset. (a) PSNR. (b) MS-SSSIM.
Fig. 7. Compressed images by JPEG2K, BPG, Li20 (MSE) [19] and Ours (MSE) on the Kodak dataset. We enlarge three patches from each image and list them below the whole image. Please zoomed-in view the images to check the details.

separately. Specially, our model follows the structure of Li20 [19] and introduce context-based nonlocal attention block for context modeling and UnetBlock for transforms. Compared to Li20 with local context-based entropy modeling, our models with the nonlocal attention block are much better, which support the effectiveness of the introduced nonlocal operation in context modeling. And at the high bit rate region, our models remarkably outperform Li20, which also indicates the effectiveness of the proposed UnetBlock in reducing the information loss brought by the transforms. Fig. 6 shows the rate-distortion curves on the Tecnick dataset, where similar trends as the Kodak dataset for both PSNR and MS-SSIM can be observed.

C. Visual Quality Evaluation

We further compare the decoded images by our method against Li20 [19], JPEG2K, and BPG in visual quality and show the sample decoded images and the uncompressed images on the Kodak dataset in Fig. 7 and Tecnick dataset in Fig. 8, respectively.

For images at low bit rate, the methods optimized with MS-SSIM are visually much better due to that MS-SSIM takes structural similarity in different scale into account and are more consistent with the human visual system. When it comes to the images at a high bit rate, the methods optimized with MSE are better at keeping small-scale details. Thus, we compare Ours(MS-SSIM) with Li20(MS-SSIM) at low bit rates as shown in Fig. 8 and compare Ours(MSE) with Li20(MSE) at higher bit rates in Fig 7. In Fig. 8, JPEG2K and BPG exhibit artifacts (such as blocking, ringing, blurring, and aliasing) that are common to all handcrafted transform coding methods. Li20(MS-SSIM) is effective at suppressing most of the artifacts but still suffers from blurring in some parts of the image. In contrast, our method optimized for MS-SSIM is more able to keep the details and has less visible distortions. In Fig. 7, the whole visual quality is nearly the same. But when zooming into details, similar to BPG, our method optimized for MSE shows to have better small scale edges and textures. Li20(MSE) blurs the small edges and loses important information, such as the text.

D. Entropy Model

The proposed context-based nonlocal entropy model exploits both the global content similarity and the local representations of the context for entropy modeling. In this section, we conduct several experiments to evaluate the performance of the proposed context-based nonlocal entropy model. First, we directly compare the context-based nonlocal entropy model with the state-of-art entropy models used in image compression. Second, we visualize the entropy of the code maps predicted by entropy models with and without nonlocal attention block and analyze the superiority of proposed special nonlocal operation. Third, we visualize the best matching code pairs found by our context-based nonlocal operation. Finally, we discuss the effect of code patch size on the nonlocal operation.

1) Comparison on Different Entropy Models: We fix the encoder and the codes produced by encoder and only test different entropy models, i.e., CABAC, the pure hyper-prior entropy model [11], the joint hyper-prior and autoregressive prior entropy model in [12], pure context-based entropy model [19] (CCN), and our nonlocal context-based entropy model (NonlocalCCN), on predicting the entropy of the same set of codes. Both the codes of the 14 models optimized for the MSE and MS-SSIM are adopt in evaluating these entropy models. CABAC is a traditional method which contains a very small context of two codes. The pure hyper-prior model further compresses the codes into hyper-prior and make entropy
Fig. 8. Compressed images by JPEG2K, BPG, Li20 (MS-SSIM) [19] and Ours (MS-SSIM) on the Tecnick dataset. We enlarge three patches from each image and list them below the whole image. Please zoomed-in view the images to check the details.

Fig. 9. Rate-distortion curves of different entropy models on the Kodak dataset. (a) PSNR. (b) MS-SSIM.

prediction with the hyper-prior. The joint entropy model combines hyper-prior with a small context of $5 \times 5$. The pure context-based entropy model adopts CCNs for context modeling and could exploit much larger context but still focuses on local representations. Compared to the pure context-based entropy model, our nonlocal entropy model could exploit the whole context and content similarity information in the context. As shown in Fig. 9, the joint model with a small context model has a clear improvement on the performance compared to the pure hyper-prior model. And another pure context model CCN with a larger receptive filed of context shows to have comparable and slightly better performance compared to the joint model. Thus, the context contribute to the most of the performance in predicting the entropy of the code at least in our model. Our nonlocal context-based entropy model overwhelms the traditional CABAC and clearly outperforms the joint entropy model of [12] and pure context-based entropy model from [19], which strongly supports the effectiveness of the proposed nonlocal attention block in context modeling.

2) Visualization of Entropy Estimation With and Without Nonlocal Attention Block: We further visualize the estimated probability for each code plane by mapping $P(y_{p,q})$ to an integer in the range of $[0, 255]$ and show the code plane as a gray image in Fig. 10. Fig. 10(a) gives the uncompressed image; (b)–(f) are the visualized probability predicted by the local entropy model on 8 code planes; (g)–(k) are the corresponding results of the context-based nonlocal entropy model. In bottom code planes, as shown in Fig. 10(b), (c), (g) and (h), CCN-based entropy model has similar performance as the proposed nonlocal entropy model. This can be illustrated by the proxy similarity function. The similarity between two codes is estimated by the proxy code vectors in the bottom code planes. With a few numbers of bottom code planes, the similarity is not accurate and thus leads to poor nonlocal estimation. The attention block tends to focus on the local representations instead of the nonlocal estimation. With the increase of planes in the bottom, the proxy similarity begins more accurate and the nonlocal estimations star to contribute to the performance.
As shown in Fig. 10(d)–(f) and (i)–(k), the context-based nonlocal entropy modeling shows to have better performance in the top code planes and reduce the code size by 27.5%, 16.9%, and 15.7% in the 28th, 30th, and 31th code plane. Especially in the region marked with green rectangles, the context-based nonlocal entropy modeling overwhelms the CCN-based local entropy modeling.

3) Visualization of the Best Matching Code Pairs: We visualize the code and its best and second best matching codes found by our method. In Fig. 11, we project the location of the code back to the image space and mark the position of the code with a 8 × 8 rectangle. The red rectangles mark the position of the target code; the blur rectangles mark the position of the best matching codes; the green rectangles show the position of the second best matching codes. As we can see, the image patches at the position marked by green and blue rectangles are quite similar as those marked by the red rectangles. This supports our claim that there should be a close relationship between codes produced by similar image patches and proves the effectiveness of the proposed proxy similarity metric.

In addition, in visualization of the matching codes, we find that the smoothness of the image would affect the performance of nonlocal operation. When the image is smooth, the code map is also smooth. For most of the codes, the best matching codes found by our context-based nonlocal operation falls on a small windows around the target, i.e., 5 × 5 windows as shown in the last row of Fig. 11. In such case, the prediction results of nonlocal operation are consistent with the local context models, such as masked convolution and CCNs. The contribution of nonlocal operation is limited. In those images consist of few smooth regions, our nonlocal model could find more similar codes in long range and reduce the entropy by a lot as shown in the first two rows of Fig. 11. This can be witnessed on the results of Kodak dataset and Tecnick dataset. The images in Kodak dataset are more complex than those of the Tecnick dataset. Thus, the improvement brought by the nonlocal operation on Tecnick dataset is not as large as that on the Kodak dataset.

4) Effect of Code Patch Size on Nonlocal Operation: For the nonlocal operation, the range of code search should be the whole code map. However, the larger code map will lead to larger memory consumption. For a code map with the size of $h \times w$, we need to calculate a weight matrix with the size of $h \times w \times h \times w$ for each code map. With the limitation of the hardware, i.e., the memory of GPUs, we cut the code block into code patches with the size of 60. We test the effect of code patch size on the entropy prediction. Patch sizes from 10 × 10 to 100 × 100 are tested in the experiments. The results are
given in Table I. The 14 pretrained models (seven for MSE and seven for MS-SSIM) are adopt in experiments. We fix the codes and only explore the effect of code patch size on entropy estimation. We adopt bits per code (bpc), i.e., the total number of bits used to represent the codes divided by the total number of codes, to evaluate the compression rate of different code patch settings. Smaller bpc indicate better performance and larger compression ratio. Generally, larger patch size do contribute to better entropy estimation performance. And the selected patch size of 60 is a good tradeoff between performance and computation consumption. When the patch size is larger than 60, the performance improvement is limited.

**E. Ablation Experiment for UnetBlock**

The width of the transforms, i.e., the number of feature maps in the output of each layer, is supposed to have a significant influence on the performance of low distortion image compression models. A narrow transform will inevitably lose some information and thus introduce extra distortion in reconstructing the input image. In this article, we conduct an experiment to support this claim by replacing the DenseBlock adopted in [19] with the proposed UnetBlock and comparing the two models, i.e., CCN+DenseBlock and CCN+UnetBlock. The width for the encoder and decoder with UnetBlock and DenseBlock are separately 196 and 64. As shown in Fig. 12, at a low bit rate region (< 0.4 bpp), UnetBlock, and DenseBlock have nearly the same performance in both of the MS-SSIM and MSE. But when the bit rate grows, the UnetBlock has a significant improvement, which supports the width of the transforms is important at high bit rate regions where the distortion is quite small. Similar trends can be observed for our model optimized with MS-SSIM.

With the same width and depth, Unet is usually fast in speed and needs less computational resources. To evaluate the performance of the UnetBlock in building the encoders and decoders, we further compare the UnetBlock to residual block [51] on a computer with an Intel(R) Xeon(R) Processor E5-2620 v4, 64 GB of RAM and a NVIDIA TITAN Xp GPU. We adopt several residual blocks to replace a single UnetBlock to keep the depth of the competing transforms. The DenseBlocks in Li20 [19] concatenate all the output of previous sub-blocks as the input for the next sub-block. The computational consumption increases dramatically with the growth of the width. Limited by the GPU memory, we fail to compare to transforms based on DenseBlock with a width of 192. Thus, we only adopt the transforms with a width of 64 for DenseBlock in comparison. Without the limitation of the entropy model ($\lambda = \infty$), the three competing models are optimized for the MSE distortion loss. Table II gives the results of the three structures on running time in seconds, GPU memory in GB and distortion performance evaluated by PSNR. The transforms based on the ResidualBlock can not process the whole image from the Kodak dataset due to the GPU memory. We only report the performance on 256 x 256 patches sampled from the Kodak dataset. The UnetBlock performs on a par with ResidualBlock on distortion performance and overwhelm the narrow network, i.e., DenseBlock. When it comes to efficiency, the transforms based on UnetBlock is much faster and needs less GPU memory than the transforms based on ResidualBlock. Thus, the UnetBlock is a good tradeoff between the speed and performance.

**F. Running on Different Architectures**

We conduct experiment to analysis the possibility of running the network on different architectures, i.e., encoding with GPU and decoding with CPU. This is a common issue for models with DNNs. There does exist small difference among the results of different hardware architectures. The main reason is different implements of the basic operation, such as convolution. One may adopt the CBLAS library or Intel MKL library for computation on CPU but have to
adopt CUBLAS developed by NVIDIA on GPU. Different implementations will lead to a small difference in the results, e.g., the average difference is less than $10^{-5}$ between the CPU and GPU in our case. In arithmetic coding, the probability table of each code is adopted to divide the range for encoding and decoding. For a single code, such small difference of the probability table would not effect the results. In a long run, with the accumulation of such difference, it would still lead to decoding errors. If the model is adopted on different hardware architectures, we suggest to divide the codes into different groups and add check codes at the end of each group. The accumulated difference could be removed after decoding one group of codes in this way.

VI. CONCLUSION

In this article, we introduce a novel proxy similarity metric and corresponding nonlocal operation for context modeling of the learned image compression methods. And a nonlocal attention block is proposed to combine the local and global representations of the context for more accurate entropy modeling. As for the network structure, we introduce an effective and efficient network structure, i.e., UnetBlock, to build the analysis transform and synthesis transform to reduce the information loss and boost the performance for low distortion compression. We test the context-based nonlocal entropy model and the UnetBlock for lossy image compression. Both of our models optimized for MSE and MS-SSIM can get satisfactory results compared to the state-of-the-art image compression standards and recent DNN-based models.

ACKNOWLEDGMENT

The authors would like to thank the NVIDIA Corporation for donating a TITAN Xp GPU used in this research.

REFERENCES

[1] G. K. Wallace, “The JPEG still picture compression standard,” IEEE Trans. Consum. Electron., vol. 38, no. 1, pp. 18–34, Feb. 1992.
[2] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, “Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising,” IEEE Trans. Image Process., vol. 26, no. 7, pp. 3142–3155, Jul. 2017.
[3] J. Xie, L. Xu, and E. Chen, “Image denoising and inpainting with deep neural networks,” in Proc. Neural Inf. Process. Syst, 2012, pp. 341–349.
[4] C. Dong, C. C. Loy, K. He, and X. Tang, “Image super-resolution using deep convolutional networks,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 38, no. 2, pp. 295–307, Feb. 2016.
[5] A. Majumdar, “Blind denoising autocoder,” IEEE Trans. Neural Netw. Learn. Syst., vol. 30, no. 1, pp. 312–317, Jan. 2018.
[6] W. Hou, X. Gao, D. Tao, and X. Li, “Blind image quality assessment via deep learning,” IEEE Trans. Neural Netw. Learn. Syst., vol. 26, no. 6, pp. 1275–1286, Jun. 2014.
[7] K. Ma, W. Liu, K. Zhang, Z. Duanmu, Z. Wang, and W. Zuo, “End-to-end blind image quality assessment using deep neural networks,” IEEE Trans. Image Process., vol. 27, no. 3, pp. 1202–1213, Mar. 2018.
[8] W. Luo, J. Li, J. Yang, W. Xu, and J. Zhang, “Convolutional sparse autoencoders for image classification,” IEEE Trans. Neural Netw. Learn. Syst., vol. 29, no. 7, pp. 3289–3294, Jul. 2017.
[9] P. Tang, X. Wang, B. Shi, X. Bai, W. Liu, and Z. Tu, “Deep FisherNet for image classification,” IEEE Trans. Neural Netw. Learn. Syst., vol. 30, no. 7, pp. 2244–2250, Jul. 2018.
[10] C. E. Shannon, “A mathematical theory of communication,” Bell Syst. Tech. J., vol. 27, no. 3, pp. 379–423, Jul./Oct. 1948.
[11] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, “Variational image compression with a scale hyperprior,” in Proc. Int. Conf. Learn. Represent., 2018.
[12] D. Minnen, J. Ballé, and G. D. Toderici, “Joint autoregressive and hierarchical priors for learned image compression,” in Proc. Neural Inf. Process. Syst., 2018, pp. 10794–10803.
[13] D. Marpe, H. Schwarz, and T. Wiegand, “Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard,” IEEE Trans. Circuits Syst. Video Technol., vol. 13, no. 7, pp. 620–636, Jul. 2003.
[14] T. Mikolov, M. Karafiát, L. Burget, J. Černocký, and S. Khudanpur, “Recurrent neural network based language model,” in Proc. Int. Speech Commun. Assoc., 2010.
[15] M. Sundermeyer, R. Schlüter, and H. Ney, “LSTM neural networks for language modeling,” in Proc. Int. Speech Commun. Assoc., 2012.
[16] A. V. D. Oord, N. Kalchbrenner, and K. Kavukcuoglu, “Pixel recurrent neural networks,” in Proc. Int. Conf. Mach. Learn., 2016, pp. 1747–1756.
[17] A. V. D. Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu, “Conditional image generation with PixelCNN decoders,” in Proc. Neural Inf. Process. Syst, 2016, pp. 4797–4805.
[18] 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.
[19] M. Li, K. Ma, J. You, D. Zhang, and W. Zuo, “Efficient and effective context-based convolutional entropy modeling for image compression,” IEEE Trans. Image Process., vol. 29, pp. 5900–5911, 2020.
[20] N. Ahmed, T. Natarajan, and K. R. Rao, “Discrete cosine transform,” IEEE Trans. Comput., vol. C-23, no. 1, pp. 90–93, Jan. 1974.
[21] O. Rippel and L. Bourdev, “Real-time adaptive image compression,” in Proc. Int. Conf. Mach. Learn., 2017, pp. 2922–2930.
[22] J. Ballé, V. Laparra, and E. P. Simoncelli, “End-to-end optimized image compression,” in Proc. Int. Conf. Learn. Represent., 2017.
[23] L. Theis, W. Shi, and F. Huszár, “Lossy image compression with compressive autoencoders,” in Proc. Int. Conf. Learn. Represent., 2017.
[24] E. Agustsson et al., “Soft-to-hard vector quantization for end-to-end learning compressible representations,” in Proc. Neural Inf. Process. Syst, 2017, pp. 1141–1151.
[25] G. Toderici et al., “Full resolution image compression with recurrent neural networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 5435–5443.
[26] M. Li, W. Zuo, S. Gui, 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.
[27] G. Toderici et al., “Variable rate image compression with recurrent neural networks,” 2015, arXiv:1511.06085. [Online]. Available: http://arxiv.org/abs/1511.06085
[28] N. Johnston et al., “Improved lossy image compression with pruning and spatially adaptive bit rates for recurrent networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 4385–4393.
[29] I. Goodfellow et al., “Generative adversarial nets,” in Proc. Neural Inf. Process. Syst, 2014, pp. 2672–2680.
[30] A. Buades, B. Coll, and J.-M. Morel, “A non-local algorithm for image denoising,” in Proc. IEEE Comput. Soc. Conf. Vis. Pattern Recognit., vol. 2, Jun. 2005, pp. 60–65.
[31] X. Wang, R. B. Girshick, A. Gupta, and K. He, “Non-local neural networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2018, pp. 7794–7803.
[32] D. Liu, B. Wen, Y. Fan, C. C. Loy, and T. S. Huang, “Non-local recurrent network for image restoration,” in Proc. Neural Inf. Process. Syst, 2018, pp. 1673–1682.
[33] Y. Zhang, K. Li, K. Li, B. Zhong, and Y. Fu, “Residual non-local attention networks for image restoration,” in Proc. Int. Conf. Learn. Represent, 2019.
[34] L. Zhuang, X. Fu, M. K. Ng, and J. M. Bioucas-Dias, “Hyperspectral image denoising based on global and nonlocal low-rank factorizations,” IEEE Trans. Geosci. Remote Sens., early access, Jan. 8, 2021, doi: 10.1109/TGRS.2020.3046038.
[35] L. Zhuang, R. Song, B. Du, and Y. Zhang, “Nonlocal low-rank tensor completion for visual data,” IEEE Trans. Cybern., vol. 51, no. 2, pp. 673–685, Feb. 2021.
[36] 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.
[37] T. Chen, H. Liu, Z. Ma, Q. Shen, X. Cao, and Y. Wang, “End-to-end learnt image compression via non-local attention optimization and improved context modeling,” IEEE Trans. Image Process., vol. 30, pp. 3179–3191, 2021.
Mu Li received the B.C.S. degree in computer science and technology from Harbin Institute of Technology, Shenzhen, China, in 2015, and the Ph.D. degree from the Department of Computing, Hong Kong Polytechnic University, Hong Kong, China, in 2020. He is the owner of Hong Kong Ph.D. Fellowship. He is currently a Post-Doctoral Researcher with the School of Data Science, the Chinese University of Hong Kong, Shenzhen. His research interests include deep learning, image processing, and image compression.

Kai Zhang received the Ph.D. degree from the School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, China, in 2019. He was a Research Assistant from July 2015 to July 2017 and from July 2018 to April 2019 with the Department of Computing of The Hong Kong Polytechnic University. He is currently a Post-Doctoral Researcher at Computer Vision Lab, ETH Zurich, Switzerland, working with Prof. Luc Van Gool and Dr. Radu Timofte. His research interests include machine learning and image processing.

Jinxing Li received the B.Sc. degree from the Department of Automation, Hangzhou Dianzi University, Hangzhou, China, in 2012, the M.Sc. degree from the Department of Automation, Chongqing University, Chongqing, China, in 2015, and the Ph.D. degree from the Department of Computing, Hong Kong Polytechnic University, Hong Kong, China, in 2019. He worked at The Chinese University of Hong Kong, Shenzhen, China, from 2019 to 2021. He is currently with Harbin Institute of Technology, Shenzhen. His research interests are pattern recognition, deep learning, medical biometrics, and machine learning.

Wangmeng Zuo (Senior Member, IEEE) received the Ph.D. degree in computer application technology from Harbin Institute of Technology, Harbin, China, in 2007. From 2004 to 2006, he was a Research Assistant with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong. From 2009 to 2010, he was a Visiting Professor with Microsoft Research Asia. He is currently a Professor with the School of Computer Science and Technology, Harbin Institute of Technology. He has published over 60 papers in top-tier academic journals and conferences. His current research interests include image enhancement and restoration, weakly supervised learning, visual tracking, and image classification. He has served as a Tutorial Organizer in ECCV 2016, an Associate Editor of the IET Biometrics, and the Guest Editor of Neurocomputing. Pattern Recognition, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, and the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS.

Radu Timofte (Member, IEEE) received the Ph.D. degree in electrical engineering from the KU Leuven, Leuven, Belgium, in 2013. From 2013 to 2016, he was a Post-Doctoral Researcher with the Computer Vision Lab, ETH Zurich, Zurich, Switzerland. Since 2016 he has been Group Leader and Lecturer in the same lab. His current research interests include deep learning, implicit models, compression, tracking, restoration and enhancement.

Dr. Timofte is an editorial board member of top journals such as IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, Elsevier Neurocomputing, Elsevier Computer Vision and Image Understanding, SIAM Journal on Imaging Sciences and served(s) as an area chair for top conferences such as CVPR 2021, IJCAI 2021, ECCV 2020, ACCV 2020, ICCV 2019. His work received several awards. He is a co-founder of Merantix and a co-organizer of NTIRE, CLIC, AIM, and PIRM events.

David Zhang (Life Fellow, IEEE) received the Graduation degree in computer science from Peking University, Beijing, China, and the M.Sc. degree in computer science and the Ph.D. degree from Harbin Institute of Technology (HIT), Harbin, China, in 1982 and 1985, respectively. In 1994, he received the second Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada. From 1986 to 1988, he was a Post-Doctoral Fellow with Tsinghua University and then an Associate Professor at the Academia Sinica, Beijing. He is currently a Chair Professor at the Hong Kong Polytechnic University and the Chinese University of Hong Kong (Shenzhen). He also serves as a Visiting Chair Professor at Tsinghua University, Beijing, and an Adjunct Professor at Peking University, Shanghai Jiao Tong University, Shanghai, China, HIT, and the University of Waterloo. His research interests are medical biometrics and pattern recognition.