

LVQAC: Lattice Vector Quantization Coupled with Spatially Adaptive Companding for Efficient Learned Image Compression

Xi Zhang

Shanghai Jiao Tong University

zhangxi_19930818@sjtu.edu.cn

Xiaolin Wu

McMaster University

xwu@ece.mcmaster.ca

Abstract

Recently, numerous end-to-end optimized image compression neural networks have been developed and proved themselves as leaders in rate-distortion performance. The main strength of these learnt compression methods is in powerful nonlinear analysis and synthesis transforms that can be facilitated by deep neural networks. However, out of operational expediency, most of these end-to-end methods adopt uniform scalar quantizers rather than vector quantizers, which are information-theoretically optimal. In this paper, we present a novel Lattice Vector Quantization scheme coupled with a spatially Adaptive Companding (LVQAC) mapping. LVQ can better exploit the inter-feature dependencies than scalar uniform quantization while being computationally almost as simple as the latter. Moreover, to improve the adaptability of LVQ to source statistics, we couple a spatially adaptive companding (AC) mapping with LVQ. The resulting LVQAC design can be easily embedded into any end-to-end optimized image compression system. Extensive experiments demonstrate that for any end-to-end CNN image compression models, replacing uniform quantizer by LVQAC achieves better rate-distortion performance without significantly increasing the model complexity.

1. Introduction

In the past five years, the research on end-to-end CNN image compression has made steady progress and led to the birth of a new class of image compression methods [1, 3, 4, 8, 21, 25, 26, 34–36, 40–44]. The CNN compression can now match and even exceed the rate-distortion performance of the previous best image compression methods [6, 33, 37, 45], which operate in the traditional paradigm of linear transform, quantization and entropy coding.

The advantages of the CNN approach of data compression come from the nonlinearity of its analysis and synthesis transforms of the autoencoder architecture, the end-to-end joint optimization of the nonlinear transforms, uni-

form quantization of the latent space and conditional entropy coding (context-based arithmetic coding) of quantized features.

Apparently, using uniform scalar quantizer in the above CNN image compression framework is motivated by operational expediency more than other considerations. Only at very high bit rate uniform quantization can approach the rate-distortion optimality [15]. It is very difficult to directly adopt and optimize a vector quantizer (VQ) in the end-to-end CNN architecture design for data compression, because VQ is a discrete decision process and it is not compatible with variational backpropagation that is necessary to the end-to-end CNN training. In [1], Agustsson et al. tried to circumvent the difficulty by a so-called soft-to-hard vector quantization scheme. Their technique is a soft (continuous) relaxation of discrete computations of VQ and entropy so that their effects can be approximated in the end-to-end training. However, in [1] the quantization centers are optimized along with the other modules, which make the whole system quite cumbersome and more difficult to train. In this paper, we propose a novel Lattice Vector Quantization scheme coupled with a spatially Adaptive Companding (LVQAC) mapping. LVQ can better exploit the inter-feature dependencies than scalar uniform quantization while being computationally almost as simple as the latter. Even if the features to be compressed are statistically independent, LVQ is still a more efficient coding strategy than scalar uniform quantization. This is because the former offers a more efficient covering of high-dimensional space than the latter, as proven by the theory of sphere packings, lattices and groups [11]. Moreover, to improve the adaptability of LVQ to source statistics, we couple a spatially adaptive companding mapping with LVQ. The resulting LVQAC design is computationally as simple and as amenable to the end-to-end training of the CNN compression model as in the originally proposed framework of [3].

Consequently, for any end-to-end CNN image compression models, replacing uniform quantizer by LVQAC achieves better rate-distortion performance without significantly increasing the model complexity; the simpler

the context-sensitive entropy model, the greater the performance gain. For instance, the checkerboard context model [18] has recently been proposed to speed up the decoding process, but it hurts the R-D performance slightly. By incorporating LVQAC into the end-to-end compression CNN with a checkerboard context model, we can achieve the same performance as the counterpart CNN using a far more expensive auto-regressive context model. This is because uniform scalar quantization needs a more complex model to compensate for its coding inefficiency, whereas LVQAC does not.

2. Overview of Learned Image Compression and Related Work

Ballé *et al.* [3] pioneered the end-to-end CNN model for image compression. Their network design is still shaped by the legacy of classical three-step signal compression algorithms; namely, transform, quantization and entropy coding. The greatest advantage of the CNN approach over the classical one comes from replacing linear transforms of the latter by more powerful and complex nonlinear transforms, which is the distinctive prowess of deep neural networks. Specifically, in the end-to-end image compression CNNs, an image vector $\mathbf{x} \in \mathbb{R}^N$ is mapped to a latent code space via a parametric nonlinear analysis transform, $\mathbf{y} = g_a(\mathbf{x}; \theta_g)$. Then this representation is quantized, yielding a discrete-valued vector $\hat{\mathbf{y}} = Q(\mathbf{y}) \in \mathbb{Z}^M$ which can be losslessly compressed using entropy coding algorithms such as arithmetic coding [31, 38] and transmitted as a sequence of bits. On the other side, the decoder recovers $\hat{\mathbf{y}}$ from the compressed codes, and subjects it to a parametric nonlinear synthesis transform $g_s(\hat{\mathbf{y}}; \phi_g)$ to build the reconstructed image $\hat{\mathbf{x}}$. The nonlinear parametric transform $g_a(\mathbf{x}; \theta_g)$ and $g_s(\mathbf{x}; \phi_g)$ are implemented by convolutional neural networks and the parameters θ_g and ϕ_g are learned over a large amount of natural images.

The training goal is to minimize the expected length of the bitstream as well as the expected distortion of the reconstructed image with respect to the original image, giving rise to a rate-distortion optimization problem:

$$\begin{aligned} & \text{minimize } R + \lambda \cdot D \\ & R = \mathbb{E}_{\mathbf{x} \sim p_x} [-\log_2 p_{\hat{\mathbf{y}}}(Q(g_a(\mathbf{x})))] \\ & D = \lambda \cdot \mathbb{E}_{\mathbf{x} \sim p_x} [d(\mathbf{x}, g_s(Q(g_a(\mathbf{x}))))] \end{aligned} \quad (1)$$

where λ is the Lagrange multiplier that determines the desired rate-distortion trade-off, p_x is the (unknown) distribution of source images and $p_{\hat{\mathbf{y}}}$ is a discrete entropy model. $Q(\cdot)$ represents quantization, implemented as rounding to the nearest integer in most of the end-to-end compression papers. The first term (representing rate) corresponds to the cross entropy between the marginal distribution of the latent and the learned entropy model, which is minimized

when the two distributions are identical. The second term (distortion) corresponds to the reconstructed error between the original image and the reconstructed image.

In most of the published end-to-end image compression CNNs, uniform scalar quantizer instead of vector quantizer (VQ) is adopted as the quantization function $Q(\cdot)$ in Eq. 1, That is,

$$\hat{y}_i = Q(y_i) = \text{round}(y_i) \quad (2)$$

where index i runs over all elements of the vectors, including channels and spatial locations. Note that both terms in Eq. 1 depend on the quantized values, and the derivatives of the quantization function are zero almost everywhere, causing gradient descent problem. To allow optimization via stochastic gradient descent, a relaxation technique is used to replace the quantizer with an additive i.i.d. uniform noise Δy_i , which has the same width as the quantization bins (one):

$$\tilde{y}_i = y_i + \Delta y_i, \quad \Delta y_i \in \mathcal{U}\left(-\frac{1}{2}, \frac{1}{2}\right). \quad (3)$$

Very few articles [1, 46] studied vector quantization in end-to-end image compression networks. Agustsson *et al.* [1] proposed a so-called soft-to-hard vector quantization module to bridge the nearest neighbor decision process of VQ and variational backpropagation that is necessary for the end-to-end CNN training. Their technique is a soft (continuous) relaxation of branching operation of VQ so that the VQ effects can be approximated in the end-to-end training. Zhu *et al.* [46] proposed a probabilistic vector quantization with cascaded estimation to estimate means and covariances. However, the optimization goal of [46] only contains the distortion term, ignoring the vital design requirement of coding rates.

After the pioneering work of Ballé *et al.* [3], many other end-to-end compression methods have been proposed to further improve the R-D performances by introducing more complex nonlinear transforms and more efficient context entropy models.

Rippel *et al.* [2, 30] proposed to learn the distribution of images using adversarial training to achieve better perceptual quality at extremely low bit rate. Johnston *et al.* [19] published a spatially adaptive bit allocation algorithm that efficiently uses a limited number of bits to encode visually complex image regions. Li *et al.* [23] developed a method to allocate the content-aware bit rate under the guidance of a content-weighted importance map. Some papers focused on investigating the adaptive context model for entropy estimation to achieve a better trade-off between reconstruction errors and required bits (entropy), including [4, 21, 25, 26], among which the CNN methods of [21, 26] are the first to outperform BPG in PSNR. Choi *et al.* [9] published a novel variable-rate learned image compression framework with a conditional auto-encoder. Cheng *et al.* [8] proposed

to use discretized Gaussian Mixture Likelihoods to parameterize the distributions of latent codes and achieved a more accurate and flexible entropy model. Lin *et al.* [24] proposed a novel spatial recurrent neural network for end-to-end image compression. The block based LSTM is utilized in spatial RNN to fully exploit spatial redundancy. Minnen *et al.* [27] introduced two enhancements, channel-conditioning and latent residual prediction, that lead to network architectures with better rate-distortion performance than existing context-adaptive models while minimizing serial processing. Zhang *et al.* [41] proposed a deep learning system for attention-guided dual-layer image compression (AGDL) by introducing a novel idea of critical pixel set.

More recently, He *et al.* [18] proposed a parallelizable checkerboard context model to speed up the decoding process in the end-to-end image compression. Yang *et al.* [39] proposed slimmable compressive autoencoders (SlimCAEs), where rate (R) and distortion (D) are jointly optimized for different capacities. Gao *et al.* [14] proposed to conduct neural image compression via Attentional Multi-scale Back Projection and Frequency Decomposition. Kim *et al.* [20] proposed a novel entropy model called Information Transformer (Informer) that exploits both global and local information in a content-dependent manner using an attention mechanism. He *et al.* [17] proposed the uneven channel-conditional adaptive coding, motivated by the observation of energy compaction in learned image compression. Cheng *et al.* [7] proposed to optimize the image compression algorithm to be noise-aware as joint denoising and compression to resolve the noisy image compression problem. There are some other papers [22, 28, 32] focused on the content adaptive approaches by updating the encoder-side component in the inference time to improve the coding performance.

3. Design of LVQAC

The main contribution of this work is to replace the uniform scalar quantization of feature values by the new LVQAC module outlined in the introduction, in the end-to-end design of compression CNNs. The LVQAC module is composed of a lattice vector quantizer (LVQ) and a spatially adaptive companding (AC) mapping. LVQ, also known as poor man's VQ, is computationally almost as simple as scalar uniform quantization, but it enjoys higher coding efficiency thanks to more efficient spatial covering property of LVQ [10].

In addition, we couple LVQ with another compression technique of adaptive companding to improve the adaptability of LVQ to source statistics.

3.1. Lattice Vector Quantization

Let $\{\mathbf{u}_1, \dots, \mathbf{u}_n\}$ be a set of linearly independent vectors in \mathbb{R}^n , which is the n -dimensional basis generating a lattice

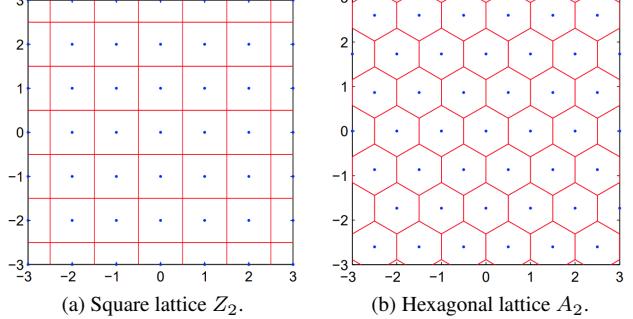


Figure 1. Two-dimensional square lattice Z_2 and hexagonal lattice A_2 and their Voronoi cells.

Λ consists of all points of the form

$$\mathbf{z} = \sum_{i=1}^n c_i \mathbf{u}_i, \quad c_i \in \mathbb{Z}. \quad (4)$$

Let $\mathbf{U} = (\mathbf{u}_1, \dots, \mathbf{u}_n)$ be the lattice generator matrix. Any vector of the lattice can be represented as $\mathbf{z} = \mathbf{c}\mathbf{U}$, $\mathbf{c} = (c_1, \dots, c_n)$, thus a lattice can be defined as:

$$\mathbf{z} \triangleq \{\mathbf{z} \in \mathbb{R}^n : \mathbf{z} = \mathbf{c}\mathbf{U}, \mathbf{c} \in \mathbb{Z}^n\}. \quad (5)$$

The Voronoi cell of a lattice point $\mathbf{z} \in \Lambda$ is defined as

$$V(\mathbf{z}) \triangleq \{\mathbf{x} \in \mathbb{R}^n : ||\mathbf{x} - \mathbf{z}|| \leq ||\mathbf{x} - \hat{\mathbf{z}}||, \forall \hat{\mathbf{z}} \in \Lambda\}. \quad (6)$$

The above structure is that of lattice vector quantization [15, 16]. The LVQ codewords are the centroids of the Voronoi polyhedrons which are of the same size and shape.

Fig. 1 depicts the concept of LVQ. It shows two examples: the two-dimensional square lattice (integer lattice) Z_2 and the hexagonal lattice A_2 , together with their Voronoi cells. The generator matrix for Z_2 lattice and A_2 lattice are as follows:

$$\mathbf{U}_{Z_2} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \mathbf{U}_{A_2} = \begin{pmatrix} 1 & 0 \\ 1/2 & \sqrt{3}/2 \end{pmatrix}. \quad (7)$$

In fact, the square lattice Z_2 is equivalent to the simple uniform scalar quantizer. The hexagonal lattice A_2 , however, offers a more efficient covering of 2D space than the square lattice Z_2 . Therefore, LVQ with the hexagonal lattice codebook offers higher coding efficiency than the square lattice codebook. It is noteworthy that the hexagonal lattice is the optimal lattice for two-dimensional space [13] but it can not be directly applied to the situation of three or higher dimensional space.

For our task of quantizing CNN feature vectors, the dimensionality of the problem is much higher than two. In the interest of low complexity and dimension independence, we choose the diamond lattice (also known as body centred lattice) instead of more complex lattice structures. The generator matrix of the n -dimensional diamond lattice is

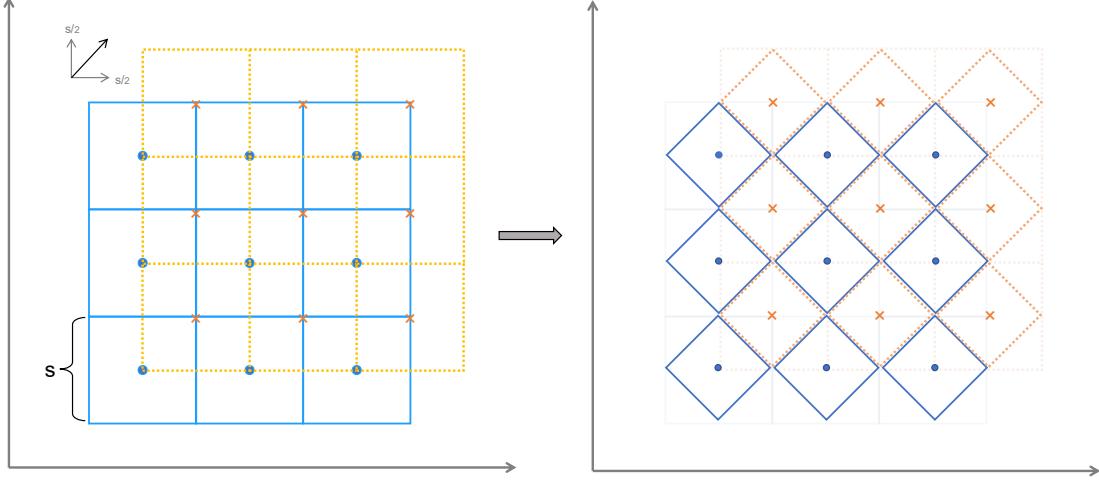


Figure 2. **Left:** Two codebooks used in diamond lattice quantization. The first codebook (blue one) is a square lattice and the second codebook (yellow one) is obtained by shifting the first codebook by $1/2$ quantization stride in each dimension. **Right:** Two-dimensional diamond lattice codebook. It can be obtained by shifting the square lattice like the left.

$$U = \begin{pmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 1/2 & -1/2 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1/2 & -1/2 & 0 & \cdots & 0 & 0 \\ \vdots & & & & \ddots & & \vdots \\ 0 & 0 & 0 & 0 & \cdots & -1/2 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 1/2 & -1/2 \end{pmatrix}. \quad (8)$$

An example of two-dimensional diamond lattice is shown in Fig. 2.

The lattice vector quantization process can be decomposed into two scalar quantizations with respective codebooks plus one argmin operation. The first codebook is a square lattice and the second codebook is obtained by shifting the first codebook by $1/2$ quantization stride in each dimension, as shown in Fig. 2. Specifically, for z to be quantized, instead of doing the regular vector quantization (computing distances between z and all centroids and select the nearest one), we do twice scalar quantizations to each element of z based on two codebooks and get two quantized vectors \hat{z}_1 and \hat{z}_2 . We choose the closer one as the lattice quantized result \hat{z} , that is

$$\hat{z} = \arg \min_{q=\{\hat{z}_1, \hat{z}_2\}} \|z - q\|. \quad (9)$$

As arg min operation is not compatible with backpropagation, we instead use its soft relaxation which is proposed in [1]. The relaxation can be written as

$$\hat{z} = \phi_a \hat{z}_1 + \phi_b \hat{z}_2. \quad (10)$$

where the weights (ϕ_a, ϕ_b) are obtained by applying soft-

max on the distances from z to \hat{z}_1 and \hat{z}_2 , that is:

$$(\phi_a, \phi_b) = \Psi(-\sigma[\|z - \hat{z}_1\|, \|z - \hat{z}_2\|]), \quad (11)$$

where Ψ is the softmax function, σ controls the degree of relaxation (or called hardness of the soft assignment as in the original paper).

Specifically, given the latent representation $y \in \mathbb{R}^{h \times w \times c}$ with c channels and $h \times w$ size, we apply the lattice quantization operation $Q_l(\cdot)$ on the latent vector for each spatial location $\{y_j \in \mathbb{R}^c, 1 \leq j \leq hw\}$, so the lattice quantized latent representation is:

$$\hat{y} = \{Q_l(y_j), y_j \in \mathbb{R}^c, 1 \leq j \leq hw\}. \quad (12)$$

It is noteworthy that in addition to transmitting the index of quantized representation, we also need to transmit extra bits to identify the selected codebook besides the quantized vectors, but the bitrate of codebook selection is negligible. As the lattice quantization is done to the latent vector for each spatial location, all elements in each spatial location share the same codebook. Thus the codebook only varies with spatial location. For a given image of size $h \times w \times 3$, the extracted latent representation will be of size $\frac{h}{16} \times \frac{w}{16} \times c$ and the codebook selection information will be of size $\frac{h}{16} \times \frac{w}{16}$ with 1-bit precision. The extra bitrate for codebook selection is less than 0.004 bpp even without binary compression.

3.2. Spatially Adaptive Companding

To further improve coding efficiency, we propose a spatially adaptive companding mapping to be coupled with the

LVQ module to boost its adaptation to source statistics. Instead of adopting a universal companding to all elements over channels and spatial locations, we choose to learn different companding functions for different spatial locations. To reduce the difficulty of learning and also guarantee the inference speed, we restrict the companding to the form of A-law function. For a given input x , the equation for A-law encoding is as follows:

$$F(x) = \text{sgn}(x) \begin{cases} \frac{A|x|}{1 + \ln(A)}, & |x| < \frac{1}{A} \\ \frac{1 + \ln(A|x|)}{1 + \ln(A)}, & \frac{1}{A} \leq |x| \leq 1 \end{cases} \quad (13)$$

where A is the scale parameter. A-law decoding is given by the inverse function:

$$F^{-1}(y) = \text{sgn}(y) \begin{cases} \frac{|y|(1 + \ln(A))}{A}, & |y| < \frac{1}{1 + \ln(A)} \\ \frac{e^{-1+|y|(1+\ln(A))}}{A}, & \frac{1}{1 + \ln(A)} \leq |x| \leq 1 \end{cases} \quad (14)$$

The A-law function and its inverse are both continuous and derivative, which implies that they can be embedded into the end-to-end optimization system.

Specifically, we learn different scale parameters A for different spatial locations and apply the unique learned A-law companding to all elements at the same location. Learning of scale parameters A is implemented by a shallow convolution network h , which consists of several convolution layers. It takes the latent representation y as input; and predicts the optimal scale parameters of A-law functions for different spatial locations, $a = h(y)$. Then we apply these learned A-law companding functions $f(F; a)$ on the latent representations to scale it to the ideal dynamic range for better quantization performance. In the other side, after decoding \hat{y} from bitstream, the inverse A-law companding $f^{-1}(F^{-1}; a)$ is applied on \hat{y} to scale it to the original dynamic range.

Given the proposed lattice vector quantization module and spatially adaptive companding mapping, the optimization goal should be rewritten to:

$$\begin{aligned} & \text{minimize } R + \lambda \cdot D \\ R &= \mathbb{E}_{\mathbf{x} \sim p_x} [-\log_2 p_{\hat{y}}(Q_l(f(g_a(\mathbf{x})))]) \\ D &= \lambda \cdot \mathbb{E}_{\mathbf{x} \sim p_x} [d(\mathbf{x}, g_s(f^{-1}(Q_l(f(g_a(\mathbf{x}))))))] \end{aligned} \quad (15)$$

where Q_l is the proposed LVQ module, f and f^{-1} are the learned spatially adaptive A-law function and its inverse.

4. Experiments

We conduct extensive experiments to systematically evaluate and analyze the effectiveness of the proposed LVQAC scheme. Specifically, we select two representative end-to-end CNN compression architectures Cheng2020 [8] and Minnen2018 [26] and replace their uniform scalar quantizers by the proposed LVQAC module to demonstrate the superiority of LVQAC. We show the R-D performance comparisons between the models with uniform scalar quantization and LVQAC for each architecture, and also provide visual comparisons. Besides, the encoding and decoding latencies are tabulated to validate the computational efficiency of LVQAC. Other analyses such as an ablation study on the effects of each component of LVQAC are further given.

4.1. Dataset and Training Details

The training dataset consists of the largest 8000 images picked from ImageNet [12], following the previous work [17]. Training images are then random-cropped to 512×512 and batched into 16. We train each model using the Adam optimizer with $\beta_1 = 0.9, \beta_2 = 0.999$. Each model is trained with an initial learning rate 10^{-4} for 2000 epochs (1M iterations), and then we decay the learning rate to 10^{-5} for another 100-epoch training. Tests involve two image sets: Kodak (24 images) and CLIC Professional valid set (41 images). The network and the proposed LVQAC module are both implemented in PyTorch [29] and CompressAI [5]. All experiments are conducted with four RTX 2080ti GPUs.

4.2. Rate-Distortion Performance

We evaluate our LVQAC scheme based on two representative published architectures Cheng2020 [8] and Minnen2018 [26]. The reason for choosing these two architectures is that they are the milestones in the end-to-end optimized image compression area, as Minnen2018 [26] is the first learning-based method to outperform BPG in PSNR metric and Cheng2020 [8] is the first work to achieve comparable performance with latest compression standard Versatile Video Coding (VVC) regarding PSNR. These two architectures were originally proposed with an autoregressive (serial) context model. To fully explore the potentiality of LVQAC, we consider the two architectures with two different context models, namely the autoregressive context model and checkerboard context model [18]. For each architecture, given two different context models, we do not change any other architectures except by replacing the uniform scalar quantizer with the proposed LVQAC scheme. We implement the two representative models coupled with uniform scalar quantizer and LVQAC, respectively. The Lagrange multipliers are set to $\lambda = \{0.0016, 0.0032, 0.0075, 0.015, 0.03, 0.045\}$.

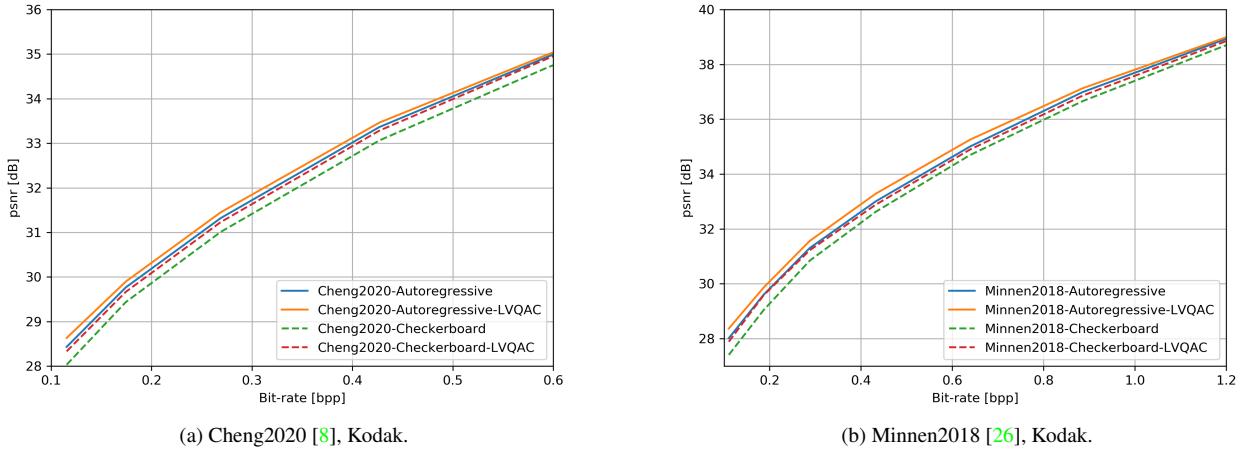


Figure 3. Rate-distortion curves on Kodak dataset. All models are optimized for MSE.

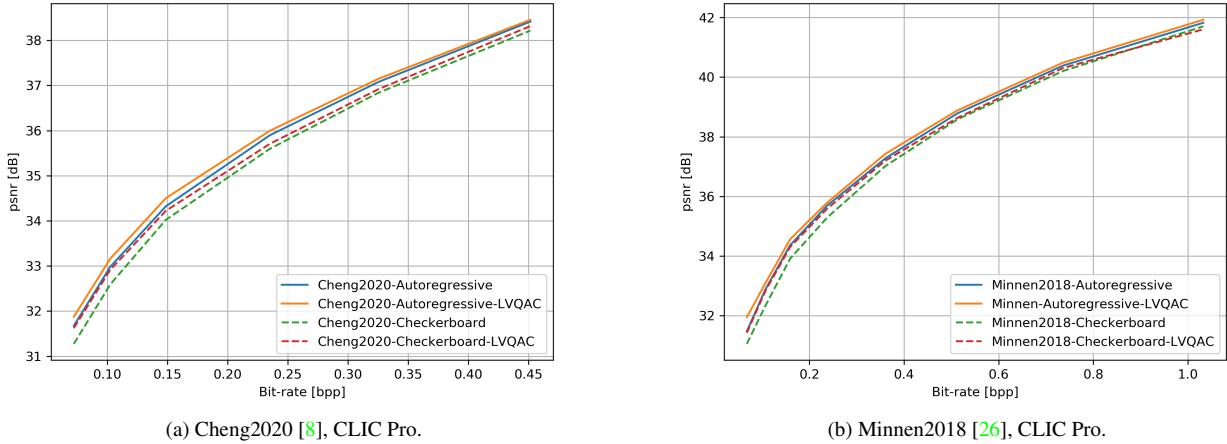


Figure 4. Rate-distortion curves on CLIC Pro dataset. All models are optimized for MSE.

Fig. 3 and Fig. 4 graphically presents the rate-distortion behaviors of different methods on Kodak and CLIC Pro datasets in PSNR metric. We can see that for any architecture, no matter with autoregressive or checkerboard context model, replacing uniform quantizer by LVQAC achieves better rate-distortion performance. Besides, LVQAC brings greater performance gain when coupled with the checkerboard context model than being coupled with the autoregressive context model. The CNN model adopted LVQAC + checkerboard context model can achieve a similar performance as the counterpart CNN using a far more expensive autoregressive context model + uniform scalar quantizer, yet the former is much lighter and faster than the latter. We show the detailed computational complexity of LVQAC in subsection 4.4. We can also find that LVQAC can bring more performance gain at the low bit rate than at the high bit rate. This is expected as uniform quantization can approach the rate-distortion optimality at very high bit rate [15].

4.3. Visual Comparison

We also provide visual comparisons to demonstrate the effectiveness of the proposed LVQAC scheme. Fig. 5, 6 and 7 show the visual results of different methods, including BPG, Cheng2020 with scalar quantizer, Cheng2002 with LVQAC and the Ground Truth. We pick three images 'kodim01.png', 'kodim07.png' and 'kodim10.png' from Kodak dataset to show the comparisons. The bit rates of all methods are adjusted to the same level for the fairness. In Fig. 5, LVQAC can preserve the sharper edges on doors than uniform scalar quantizer. In Fig. 6, LVQAC also achieves richer and clearer textures on flowers over uniform scalar quantizer. Another interesting observation is that the faint lines in Fig. 7 are both removed by BPG and scalar quantizer, but LVQAC can preserve such features. These observations reveal that the end-to-end CNN compression model coupled with LVQAC can produce much sharper edges and

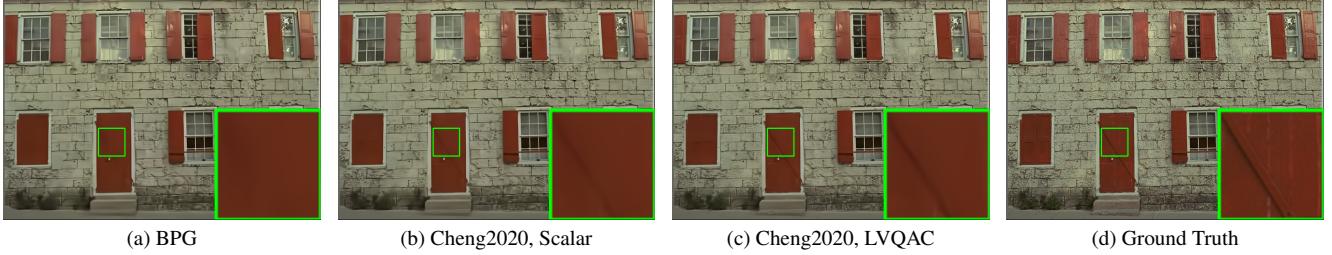


Figure 5. Visual comparisons of different methods on 'kodim01.png' from Kodak dataset.

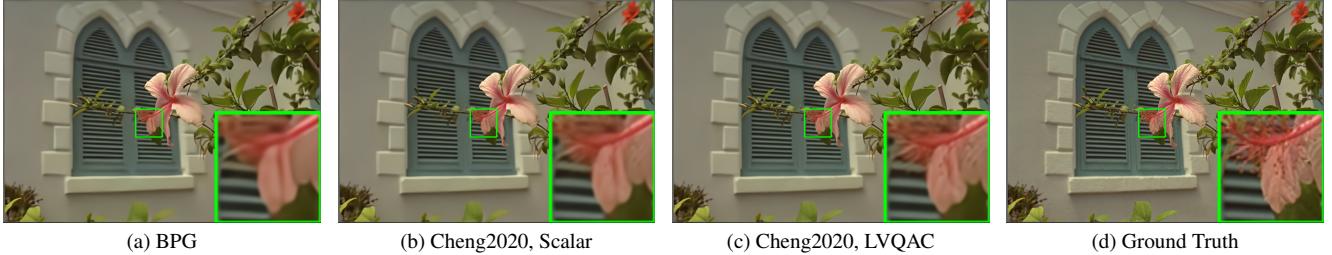


Figure 6. Visual comparisons of different methods on 'kodim07.png' from Kodak dataset.

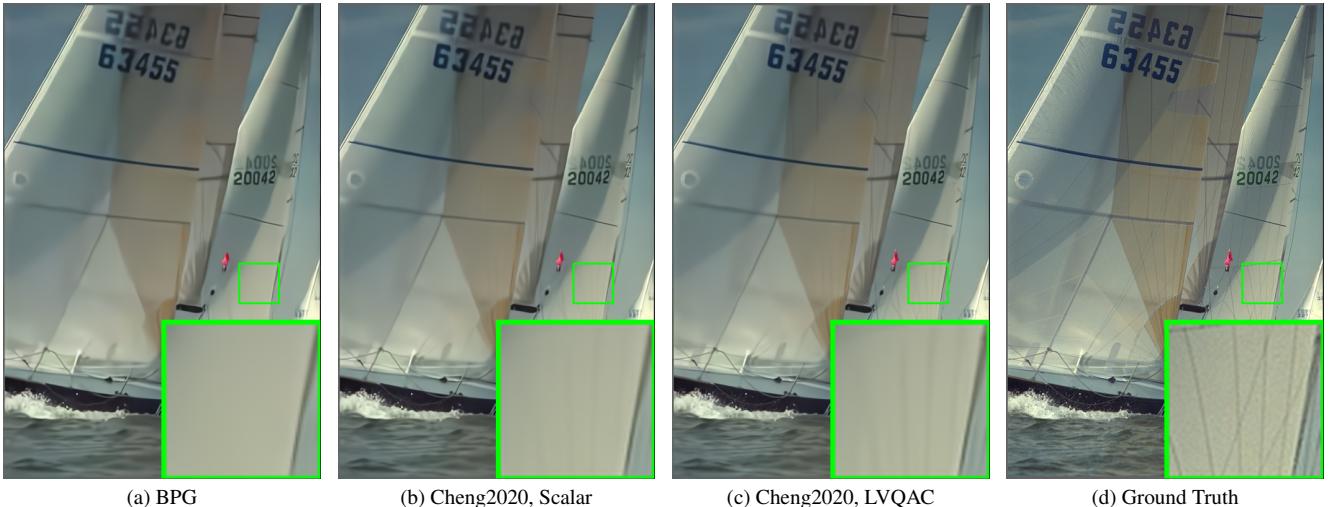


Figure 7. Visual comparisons of different methods on 'kodim10.png' from Kodak dataset.

clearer textures than the counterpart CNN coupled with scalar quantizer. We present more high-resolution visual examples in the supplementary material.

4.4. Encoding and Decoding Latency

We provide a detailed analysis of encoding and decoding latency in this subsection. Specifically, we evaluate the inference latency of end-to-end CNN compression models with different quantizers: uniform scalar quantizer and the proposed LVQAC. Table 1 tabulates the encoding time and decoding time of different architectures coupled with scalar quantizer and LVQAC. We can see that, for the ar-

chitecture Cheng2020 with checkerboard context, LVQAC only increases the encoding time by about 15% and the decoding time by about 5%, yet improves the performance to the same level as that using autoregressive context. It is noteworthy that the decoding time of the model using autoregressive context is two orders of magnitude higher than that of the checkerboard context. For the architecture Cheng2020 with autoregressive context model, the increase of encoding time is about 17%, and the increase of decoding time is negligible due to the huge base ($> 10^3$). Similar observations can be obtained on the Minnen2018 architecture. These observations reveal that replacing uniform quantizer

Table 1. BD-Rates and inference times of different methods. The BD-Rate data is calculated relative to VVC (YUV 444) from PSNR-BPP curve on Kodak. [A] represents the autoregressive context model, [C] represents the checkerboard context model.

Architecture	Context + Quantizer	BD-Rate (%)	Enc. Time (ms)	Dec. Time (ms)
Cheng2020 [8]	[A] + LVQAC	3.12	48.65	$> 10^3$
	[A] + Scalar	3.35	41.42	$> 10^3$
	[C] + LVQAC	3.39	52.66	55.84
	[C] + Scalar	3.89	45.43	53.01
Minnen2018 [26]	[A] + LVQAC	14.75	25.87	$> 10^3$
	[A] + Scalar	14.92	18.14	$> 10^3$
	[C] + LVQAC	16.17	26.23	27.10
	[C] + Scalar	20.00	18.84	24.52

by LVQAC can achieve better rate-distortion performance without significantly increasing the model complexity; the simpler the context-sensitive entropy model, the greater the performance gain.

4.5. Ablation Study

To investigate the impacts of the proposed LVQAC scheme, we conduct an ablation study to investigate the importance of each component in the proposed LVQAC scheme. We experiment progressively to explore the respective contribution of the lattice vector quantization module and spatially adaptive companding module. Specifically, we progressively add LVQ and AC modules into the network and test the coding performance. Table 2 shows the results of the ablation study. It can be seen that lattice vector quantization and spatially adaptive companding contribute almost equally to the performance gain.

4.6. Limitation

The main limitation of this work relies on that it is difficult to know the optimal lattice structure for any given dimension. In this paper, we choose the diamond lattice structure due to its simplicity and easy extension to any dimension. However, even though diamond lattice is not the optimal lattice structure in the vast majority of cases, it is still much more efficient than uniform scalar quantizer without significantly increasing the model complexity. Another limitation of LVQAC is that if the context model is complex and powerful enough, such as an autoregressive context, then the gain brought by LVQAC will be very limited. Thus the application of LVQAC may be restricted to those lightweight end-to-end CNN compression models.

5. Conclusion and Future Work

In this paper, we present a novel Lattice Vector Quantization scheme coupled with a spatially Adaptive Companding (LVQAC) mapping. LVQ can better exploit the inter-feature dependencies than scalar uniform quantization while being

Table 2. Results of the progressive ablation study. The BD-Rate data is calculated relative to VVC (YUV 444) from PSNR-BPP curve on Kodak. [A] represents the autoregressive context model, [C] represents the checkerboard context model.

Architecture	Context + Quantizer	BD-Rate
Cheng2020 [8]	[A] + LVQ + AC	3.12
	[A] + LVQ	3.24
	[A] + Scalar + AC	3.30
	[A] + Scalar	3.35
Minnen2018 [26]	[C] + LVQ + AC	3.39
	[C] + LVQ	3.68
	[C] + Scalar + AC	3.82
	[C] + Scalar	3.89
	[A] + LVQ + AC	14.75
	[A] + LVQ	14.86
	[A] + Scalar + AC	14.88
	[A] + Scalar	14.92
	[C] + LVQ + AC	16.17
	[C] + LVQ	18.43
	[C] + Scalar + AC	19.29
	[C] + Scalar	20.00

computationally almost as simple as the latter. Moreover, to improve the adaptability of LVQ to source statistics, we couple a spatially adaptive companding (AC) mapping with LVQ. The resulting LVQAC design can be easily embedded into any end-to-end optimized image compression system. Extensive experiments demonstrate that for any end-to-end CNN image compression models, replacing uniform quantizer by LVQAC achieves better rate-distortion performance without significantly increasing the model complexity. Future work will focus on exploring the optimal lattice vector quantizer for a given dimension and how to embed it into the end-to-end image compression system.

References

- [1] Eirikur Agustsson, Fabian Mentzer, Michael Tschannen, Lukas Cavigelli, Radu Timofte, Luca Benini, and Luc Van Gool. Soft-to-hard vector quantization for end-to-end learning compressible representations. In *Advances in Neural Information Processing Systems 30*, pages 1141–1151, 2017. 1, 2, 4
- [2] Eirikur Agustsson, Michael Tschannen, Fabian Mentzer, Radu Timofte, and Luc Van Gool. Generative adversarial networks for extreme learned image compression. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 221–231, 2019. 2
- [3] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. End-to-end optimized image compression. In *5th International Conference on Learning Representations, ICLR*, 2017. 1, 2
- [4] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. In *6th International Conference on Learning Representations, ICLR*. OpenReview.net, 2018. 1, 2
- [5] Jean Bégaint, Fabien Racapé, Simon Feltman, and Akshay Pushparaja. Compressai: a pytorch library and evaluation platform for end-to-end compression research. *arXiv preprint arXiv:2011.03029*, 2020. 5
- [6] Fabrice Bellard. Better portable graphics (bpg). <https://bellard.org/bpg/>. 1
- [7] Ka Leong Cheng, Yueqi Xie, and Qifeng Chen. Optimizing image compression via joint learning with denoising. In *European Conference on Computer Vision*, pages 56–73. Springer, 2022. 3
- [8] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. Learned image compression with discretized gaussian mixture likelihoods and attention modules. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pages 7936–7945, 2020. 1, 2, 5, 6, 8
- [9] Yoojin Choi, Mostafa El-Khamy, and Jungwon Lee. Variable rate deep image compression with a conditional autoencoder. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3146–3154, 2019. 2
- [10] J Conway and N Sloane. Fast quantizing and decoding algorithms for lattice quantizers and codes. *IEEE Transactions on Information Theory*, 28(2):227–232, 1982. 3
- [11] John H. Conway and Neil J. A. Sloane. *Sphere Packings, Lattices and Groups*, volume 290 of *Grundlehren der mathematischen Wissenschaften*. Springer, 1988. 1
- [12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 5
- [13] Lenny Fukshansky. Revisiting the hexagonal lattice: on optimal lattice circle packing. *Elemente der Mathematik*, 66(1):1–9, 2011. 3
- [14] Ge Gao, Pei You, Rong Pan, Shunyuan Han, Yuanyuan Zhang, Yuchao Dai, and Hojae Lee. Neural image compression via attentional multi-scale back projection and frequency decomposition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14677–14686, 2021. 3
- [15] Allen Gersho and Robert M Gray. *Vector quantization and signal compression*, volume 159. Springer Science & Business Media, 2012. 1, 3, 6
- [16] Robert M. Gray and David L. Neuhoff. Quantization. *IEEE transactions on information theory*, 44(6):2325–2383, 1998. 3
- [17] Dailan He, Ziming Yang, Weikun Peng, Rui Ma, Hongwei Qin, and Yan Wang. Elic: Efficient learned image compression with unevenly grouped space-channel contextual adaptive coding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5718–5727, 2022. 3, 5
- [18] Dailan He, Yaoyan Zheng, Baocheng Sun, Yan Wang, and Hongwei Qin. Checkerboard context model for efficient learned image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14771–14780, 2021. 2, 3, 5
- [19] Nick Johnston, Damien Vincent, David Minnen, Michele Covell, Saurabh Singh, Troy Chinen, Sung Jin Hwang, Joel Shor, and George Toderici. Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4385–4393, 2018. 2
- [20] Jun-Hyuk Kim, Byeongho Heo, and Jong-Seok Lee. Joint global and local hierarchical priors for learned image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5992–6001, 2022. 3
- [21] Jooyoung Lee, Seunghyun Cho, and Seung-Kwon Beack. Context-adaptive entropy model for end-to-end optimized image compression. In *7th International Conference on Learning Representations, ICLR*, 2019. 1, 2
- [22] Meng Li, Shangyin Gao, Yihui Feng, Yibo Shi, and Jing Wang. Content-oriented learned image compression. *arXiv preprint arXiv:2207.14168*, 2022. 3
- [23] Mu Li, Wangmeng Zuo, Shuhang Gu, Debin Zhao, and David Zhang. Learning convolutional networks for content-weighted image compression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3214–3223, 2018. 2
- [24] Chaoyi Lin, Jiabao Yao, Fangdong Chen, and Li Wang. A spatial rnn codec for end-to-end image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13269–13277, 2020. 3
- [25] Fabian Mentzer, Eirikur Agustsson, Michael Tschannen, Radu Timofte, and Luc Van Gool. Conditional probability models for deep image compression. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 4394–4402, 2018. 1, 2
- [26] David Minnen, Johannes Ballé, and George Toderici. Joint autoregressive and hierarchical priors for learned image compression. In *Advances in Neural Information Processing Systems 31*, pages 10794–10803, 2018. 1, 2, 5, 6, 8
- [27] David Minnen and Saurabh Singh. Channel-wise autoregressive entropy models for learned image compression. In *2020*

- IEEE International Conference on Image Processing (ICIP)*, pages 3339–3343. IEEE, 2020. 3
- [28] Guanbo Pan, Guo Lu, Zhihao Hu, and Dong Xu. Content adaptive latents and decoder for neural image compression. In *European Conference on Computer Vision*, pages 556–573. Springer, 2022. 3
- [29] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 5
- [30] Oren Rippel and Lubomir Bourdev. Real-time adaptive image compression. *arXiv preprint arXiv:1705.05823*, 2017. 2
- [31] Jorma Rissanen and Glen G Langdon. Arithmetic coding. *IBM Journal of research and development*, 23(2):149–162, 1979. 2
- [32] Chajin Shin, Hyeongmin Lee, Hanbin Son, Sangjin Lee, Dogyoon Lee, and Sangyoun Lee. Expanded adaptive scaling normalization for end to end image compression. In *European Conference on Computer Vision*, pages 390–405. Springer, 2022. 3
- [33] Athanassios Skodras, Charilaos Christopoulos, and Touradj Ebrahimi. The jpeg 2000 still image compression standard. *IEEE Signal processing magazine*, 18(5):36–58, 2001. 1
- [34] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. Lossy image compression with compressive autoencoders. In *5th International Conference on Learning Representations, ICLR*, 2017. 1
- [35] George Toderici, Sean M. O’Malley, Sung Jin Hwang, Damien Vincent, David Minnen, Shumeet Baluja, Michele Covell, and Rahul Sukthankar. Variable rate image compression with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR*, 2016. 1
- [36] George Toderici, Damien Vincent, Nick Johnston, Sung Jin Hwang, David Minnen, Joel Shor, and Michele Covell. Full resolution image compression with recurrent neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 5435–5443, 2017. 1
- [37] Gregory K Wallace. The jpeg still picture compression standard. *IEEE transactions on consumer electronics*, 38(1):xviii–xxxiv, 1992. 1
- [38] Ian H Witten, Radford M Neal, and John G Cleary. Arithmetic coding for data compression. *Communications of the ACM*, 30(6):520–540, 1987. 2
- [39] Fei Yang, Luis Herranz, Yongmei Cheng, and Mikhail G Mozerov. Slimmable compressive autoencoders for practical neural image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4998–5007, 2021. 3
- [40] Xi Zhang and Xiaolin Wu. Ultra high fidelity deep image decompression with ℓ_∞ -constrained compression. *IEEE Transactions on Image Processing*, 30:963–975, 2020. 1
- [41] Xi Zhang and Xiaolin Wu. Attention-guided image compression by deep reconstruction of compressive sensed saliency skeleton. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13354–13364, 2021. 1, 3
- [42] Xi Zhang and Xiaolin Wu. Multi-modality deep restoration of extremely compressed face videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. 1
- [43] Xi Zhang and Xiaolin Wu. Dual-layer image compression via adaptive downsampling and spatially varying upconversion. *arXiv preprint arXiv:2302.06096*, 2023. 1
- [44] Xi Zhang, Xiaolin Wu, Xinliang Zhai, Xianye Ben, and Chengjie Tu. Davd-net: Deep audio-aided video decompression of talking heads. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12335–12344, 2020. 1
- [45] Xin Zhao, Seung-Hwan Kim, Yin Zhao, Hilmi E Egilmez, Moonmo Koo, Shan Liu, Jani Lainema, and Marta Karczewicz. Transform coding in the vvc standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(10):3878–3890, 2021. 1
- [46] Xiaosu Zhu, Jingkuan Song, Lianli Gao, Feng Zheng, and Heng Tao Shen. Unified multivariate gaussian mixture for efficient neural image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17612–17621, 2022. 2