A Novel Deep Progressive Image Compression Framework

Chunlei Cai, Li Chen, Xiaoyun Zhang, Guo Lu, Zhiyong Gao
Institute of Image Communication and Network Engineering
Shanghai Jiao Tong University
Shanghai, China
{caichunlei, hilichen, xiaoyun.zhang, luguo2014, zhiyong.gao}@sjtu.edu.cn

***Abstract*—In Internet applications, compressing the image without perceptually distinguishable distortions and loading the images without notable delays in the client end can significantly improve the user experience. Compressing the image at high bit rates can maintain the high quality of the decoded image but in cost of long transmitting and decoding time, resulting in bad user experience. The progressive coding scheme can resolve the conflict between the high quality requirement and the large loading delay. This paper proposes a novel efficient progressive image coding framework based on deep convolutional neural networks. The proposed framework is composed of a uniform encoder network and two progressive decoder networks. The encoder network decomposes the input image into two scales of representations, that can be transmitted and reconstructed progressively into a basic quality preview image and a high-quality image by two individual decoder networks respectively. All the networks are jointly learned when achieving the rate distortion optimization of both scales. Experiments results show that the proposed method has much better coding performance than the commercial codecs WebP and JPEG, which are commonly used in Internet applications. Meanwhile, the proposed codec consumes much less time to load the image compared with WebP.**

***Index Terms*—High quality image compression, Progressive image coding, Convolutional neural networks, Rate distortion optimization**

I. Introduction

To satisfy the requirements of Internet users for high quality, the images are commonly coded at high bit rates, which outputs the compressed results without perceptually distortions. However, as the bit-rate gets high, both of the transmission time and the decoding complexity are also increased, resulting in large loading delay and bad user experience. To resolve the conflict between the high quality demands and the large loading delay, the progressive coding scheme is proposed [1] to code an image into several scales which can be transmitted and decoded progressively.

WebP and JEPG are the most commonly used image codecs in Internet applications for their high efficiency. But WebP doesn't support progressive coding, when the bit rate is set high, it will introduce notable decoding delays. Meanwhile, JPEG encodes and decodes images very fast and it supports progressive coding, so it is still the first choice in many applications, however its coding performance is much worse, resulting in larger bandwidth and storage consumption.

In this paper, we explorers to develop a novel progressive coding framework with both high coding performance and low loading delays. The proposed framework is based on deep learning based on convolutional neural network (CNN) inspired by recent remarkable successes on image compression [2]–[15]. Some CNN based image compression methods [6] have shown better coding performance than most of the traditional ones, including the best image codec BPG [16]. Moreover, CNNs can be performed highly parallel, which is easily accelerated by multi-core devices, such as graphic power unit (GPU). However, although deep learning based compression methods have shown great potential, most of them don't support the progressive coding [2], [4], [5], [12], hindering their application in practice. In addition, Toderici's method [11] can be considered as a progressive codec, but neither its performance nor its efficiency is far from satisfying.

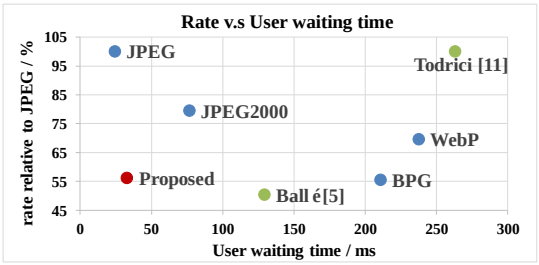

Fig. 1: Rate v.s user waiting time of different methods for high quality image compression test on Kodak (quality threshold is 40 dB PSNR).

In contrast, we proposed a novel method which has high coding performance and supports the progressive coding with high efficiency. The proposed progressive coding framework consists of a uniform encoder network and two progressive decoder networks. The encoder network decomposes the input image to generate two scales of representations simultaneously. They can be progressively reconstructed by two decoder networks into a preview image and a transparently compressed image (which means the image is coded without perceptual compression distortions) respectively. The first scale contains a small number of representations, which are entropy coded into a few bits and transmitted to the decoder end fast. The preview image is quickly reconstructed by an efficient preview decoder network within a small delay. Then the remaining image representations, is transmitted and reconstructed by

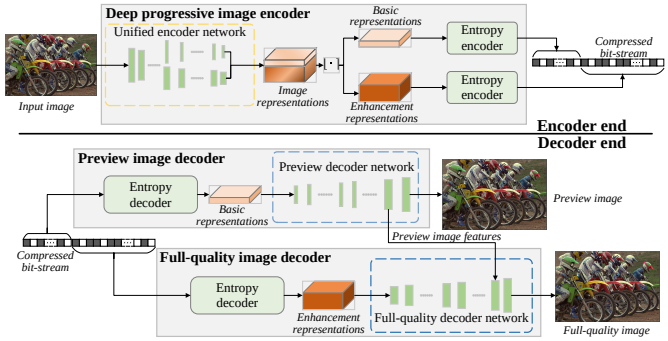

Fig. 2: The framework of the proposed progressive image codec. In the encoder end, the input image is first decomposed into a basic and an enhancement scales of representations through a unified encoder CNN. Then they are separately entropy coded into a progressive bit-stream. In the decoder ender, the bit-stream is sequentially entropy decoded into representations and they are reconstructed into a preview image and a full-quality image by two progressive decoder CNNs.

another full-quality decoder network to decode the full quality image. The entropy coding is based on CNN based distribution estimation inspired by Ballé [5]. All the networks are jointly trained to achieve the minimized rate distortion losses of both scales.

Comparative experiments on the high quality compression, as shown in Fig. 1, have demonstrated the proposed method has achieved outperforming coding performance, which is significantly higher than either WebP or JPEG2000 and competitive with BPG and state-of-the-art CNN based methods [5], [11]. Moreover, the proposed method has negligible preview image loading delays and consumes much less time on decoding the full quality image compared with WebP or Ballé's [5] method. For example, the proposed method can saves 86.11% and 62.19% time to load the preview and full-quality images respectively when 40 dB PSNR quality threshold are achieved.

II. Proposed Deep Progressive Image Compression

A. Overview of the Proposed Framework

The proposed CNN based progressive image compression framework is shown as Fig. 2. The encoder end contains a unified image encoder network. It transforms the input image into basic representations and enhancement representations simultaneously. While in the decoder end, two progressive decoder networks are included. The preview image decoder network reconstructs the basic representations into a preview image. The full-quality image decoder reconstructs the enhancement information from the enhancement representations and then use the information to compensate the preview image features into a full-quality image. Besides, two entropy codecs, which entropy context models are also generated based on deep learning, are used to code the basic and enhancement representations into the progressive bit-stream.

B. Network Architectures

We present an implementation example of the proposed progressive encoder and decoder networks, as shown in Fig. 3. Note that we use the presented architectures as we are pursuing a practical codec with real-time efficiency, however the internal structure of the networks is fairly unrestricted, e.g., one could exchange the space-to-depth (depth-to-space) layer for convolution (deconvolution) or build larger networks for potential higher coding performance without fundamentally changing the model architecture.

In this paper, the input image is transformed into basic and enhancement representations simultaneously. This is implemented by a unified encoder network inspired by the work of Cai et al. [7] in where a multi-scale decomposition module is proposed to decompose image features into scalable representations. However, in their method, all scales share the same decoder network, which means the decoder process has the constant complexity regardless the scales and thus the progressive decoding is inefficient based on their method. Different from their method which puts the decomposition module at the end of the encoder network, we decompose the image at the front end of the network and then simultaneously transform the decomposed image features into separable scalable image representations, which can be individually and progressively decoded with scaled complexity.

Specifically, the unified encoder network is mainly built by stacks of linear operations followed by non-linear normalization inspired by the method of Ballé et al. [5]. Different from the network used in [5], in this paper, we place a space-to-depth (S2D) layer followed by a convolution layer at the front end of the network to improve the execution efficiency. S2D rearranges the blocks of spatial image pixels into depth. It is a lossless operation, which outputs a copy of the input image where pixels from the height and width dimensions are moved to the depth dimension. We use this operation to reduce the spatial dimension of the input image, leading to almost halved

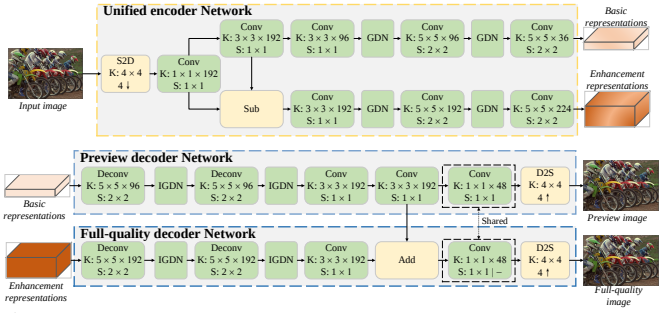

Fig. 3: An implementation example of the proposed progressive encoder/decoder networks. In the figures, the "S2D" block means space-to-depth operation, followed by the block size ("K") and the down-sample scale factor and "D2S" is the depth-to-space operation. The "Conv" blocks represent convolution layers, which kernel size ("K") and the stride size ("S") are shown below, and "Deconv" means deconvolution layer. "Sub" is element subtraction operation and "Add" is element adding operation. "GDN" and "IGDN" are generalized divisive normalization and its inverse operations [17]. "Shared" means the last two layers of the preview and the full-quality decoder networks share the same parameters. Note that all the learned operations are shown in green blocks and other fixed operations are in yellow blocks.

computational complexity compared with the encoder network proposed in [5].

The decoder end is fundamentally the counterpart of the unified encoder network, but they are separated into two progressive decoder networks. In detail, the quantized basic representations are first input into the preview decoder network to reconstruct a basic quality image. The preview decoder network is implemented as high efficient by using much fewer filters within the inverse transformation. The full-quality decoder network takes the similar architecture with the preview decoder network but with more filters. To reduce the computation redundancy, at the end of the network, the full-quality image is integrated by the preview image features and the enhancement features reconstructed from the quantized enhancement representations. Note that the last deconvolution layers of the preview decoder network and the full-quality decoder network are shared, because it is a counterpart of the first convolution layer in the unified encoder network.

C. Optimization of the Proposed Method

Once the implementation architectures of the proposed framework has been proposed, the progressive codec is determined by the network parameters, that includes the unified encoder network ($\boldsymbol{\phi}$), the progressive decoder networks ($\boldsymbol{\theta}_{pre}$, $\boldsymbol{\theta}_{enh}$) and the entropy codecs ($\boldsymbol{\omega}_{pre}$, $\boldsymbol{\omega}_{enh}$). In this paper, we optimize the proposed framework by minimizing the rate distortion (RD) costs of the preview and the full-quality image compression as defined as Eq. 1.

$$\begin{aligned}\chi^* = \arg\min_{\chi} \ \{&[R_{ful}(\chi) + \lambda_{ful} D_{ful}(\chi)] + \\ &[R_{pre}(\chi) + \lambda_{pre} D_{pre}(\chi)]\}\end{aligned} \tag{1}$$

where $\chi = \{\boldsymbol{\phi}, \boldsymbol{\theta}_{pre}, \boldsymbol{\theta}_{enh}, \boldsymbol{\omega}_{pre}, \boldsymbol{\omega}_{enh}\}$ represent all the parameters in the proposed framework and χ^* is the optimized solution corresponds to the minimized loss. The total loss function is a sum of the RD cost of the full-quality compression and the preview compression. Specifically, R_{ful} and D_{ful} are the rate and the distortion of the the full-quality compression result. R_{pre} and D_{pre} are the losses for the preview one. λ_{ful} and λ_{pre} are used to control the balance between the rate and the distortion. For the preview compression result, the rate is supposed to be low for fast transmission, and for the full-quality compression, the quality is the first prior, so λ_{pre} is set very much lower relative to λ_{ful}.

1) Distortion Measurement: In order to compare with existent coding methods for high quality compression, which are commonly optimized for mean square error (MSE), we measure the distortions of the full quality based on MSE. It is defined as Eq. 2.

$$D_{ful} = \|\boldsymbol{I} - \boldsymbol{I}_{ful}\|^2 \tag{2}$$

where $\boldsymbol{I}_{ful}$ is the full-quality decoded image. But for the preview image, which will seriously lose the information after compression, minimizing the average image distortion based on MSE will result in too smooth and blurry reconstructions [3]. Besides, PSNR is not in line with the human vision system especially at low rates. So we propose to use the multiscale structural similarity index metric (MS-SSIM) [18] to measure the quality of the preview image to save more structural information for better visual quality. The distortion of the preview image is defined as Eq. 3.

$$D_{pre} = 1 - MSSSIM(\boldsymbol{I}, \hat{\boldsymbol{I}}_{pre}) \tag{3}$$

2) Rate Estimation: The entropy coding operations, such arithmetic coding, are non-differential, so when the networks are optimized by back propagation [19], the rate term is usually estimated based on entropy. Many work has been proposed on differential rate estimation for deep image compression [2], [4], [5], [12]. In this paper, we utilize the method proposed by Ballé [5]. This distribution of the representations are estimated as defined as Eq. 4.

$$\begin{aligned}p_{\boldsymbol{F}}(\boldsymbol{F}) &= \mathcal{N}(\boldsymbol{0}, \boldsymbol{\sigma}^2) \\ &with\ \boldsymbol{\sigma} = g_s(\lfloor \boldsymbol{Z} \rceil; \boldsymbol{\psi}_s)\ and\ \boldsymbol{Z} = g_a(\boldsymbol{F}; \boldsymbol{\psi}_a)\end{aligned} \tag{4}$$

The image representations $\boldsymbol{F}$ are modeled as independent individual distributed and following zero-centered Gaussian distribution with variances $\boldsymbol{\sigma}$ represented by the hyper prior information. $\boldsymbol{Z}$ represents the hyper prior of the image, which is analyzed by an analysis network g_a at the encoder end before the image representations $\boldsymbol{F}$ are entropy coded. Then the distribution of $\boldsymbol{F}$ is estimated based on the variances $\boldsymbol{\sigma}$ reconstructed from quantized $\lfloor \boldsymbol{Z} \rceil$ by a synthesis network. So $\lfloor \boldsymbol{Z} \rceil$ should also be transmitted to the decoder as the side information. The distribution of $\boldsymbol{Z}$ is estimated differentially by a linear spline function $p_{\boldsymbol{Z}}$, which sample points are jointly updated with other parameters.

The total rate is calculated as Eq. 5.

$$\begin{aligned}R &= R_{\boldsymbol{F}} + R_{\boldsymbol{Z}} = \\ &\frac{1}{H \times W} \sum [-log_2 p_{\boldsymbol{F}}(\boldsymbol{F})] + [-log_2 p_{\boldsymbol{Z}}(\boldsymbol{Z})]\end{aligned} \tag{5}$$

In this paper, the rate includes two parts: the rate R_{pre} of the basic representations for the preview image $\boldsymbol{F}_{bas}$ and the rate R_{enh} for the enhancement representations $\boldsymbol{F}_{enh}$. R_{pre} is calculated based on input $\boldsymbol{F}_{pre}$ using Eq. 4 and Eq. 5, and R_{enh} is calculated based on $\boldsymbol{F}_{enh}$. Finally, The total rate R_{ful} of the full-quality image is the sum of R_{pre} and R_{enh}. Although We estimate R_{pre} and R_{enh} using the same method, the distribution model parameters for them are individual, which means $\boldsymbol{\omega}_{pre}$ and $\boldsymbol{\omega}_{enh}$ contains their own hyper prior analysis and synthesis network parameters and sample points of the spline function.

III. Experiments

A. Implementation Details

During training, the hyper parameters λ_{pre} and λ_{ful} can control the rate of the model. To generate models with various rates, we train eight models with λ_{im} and λ_{roi} pairs are set as $\{\lambda_{pre}, \lambda_{ful} | \lambda_{ful} = 256\lambda_{pre} = 32\alpha,\ \alpha = \{0.2, 0.3, 0.5, 0.75, 1.0, 1.5, 2.0, 2.5\}\}$. With the settings, the qualities of the full-quality image of all the models are higher

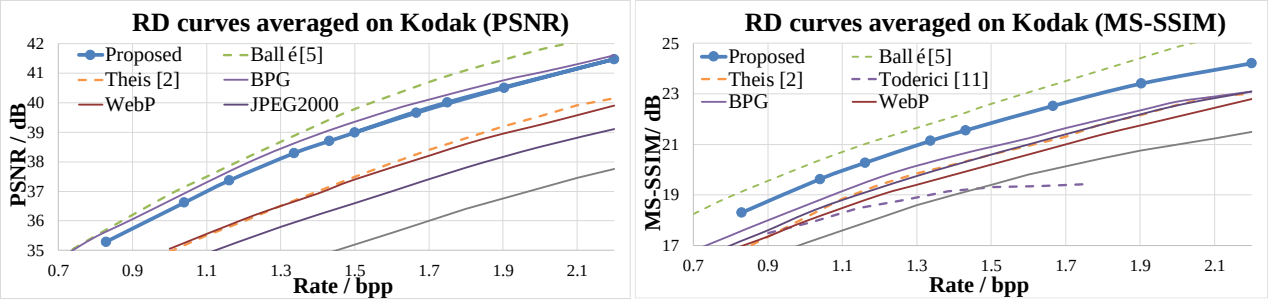

Fig. 4: Rate distortion curves of different methods averaged on Kodak. MS-SSIM is presented as $-10log_{10}(1-msssim)$ dB as Toderici et al. did in [9]. Note that the highest PSNR of Toderici's method is not larger than 35 dB, so we can't see their RD curves in terms of PSNR.

than 35 dB in terms of PSNR and the rates of the preview image are lower than 0.6 bits per pixel (bpp).

We train the proposed networks for the loss function as Eq. 1 on a large-scale set formed by 20891 high-quality natural images downloaded from flickr.com. We uniformly use the Adam optimization algorithm for training all models. To save the time cost of the training process, we first train the model with setting: $\lambda_{ful} = 8192, \lambda_{pre} = 32$, and then fine-tune other models from the pre-trained model as the scheme proposed by Thesis et al [2].

B. Comparison with other Methods

We compare our method with 4 traditional codecs (BPG [16], WebP [20], JPEG2000 [21], JPEG [22]). and 3 CNN based methods (Ballé's [5], Toderici's [11] and Theis' [2]) in terms of coding performance for high quality compression and the executing time. Among the comparative methods, Toderici's [11], JPEG and JPEG2000 can work as progressive codecs, so we also present the comparison results of their preview images.

We first present the rate distortion curves averaged on Kodak test image set [23] in Fig. 4. We set the quality threshold as 35 dB in terms of PSNR and compare the methods at the quality higher than the threshold. As we can see, the coding performance of the proposed method is significantly better than either JPEG2000 or WebP, and competitive to the best traditional codec (BPG) and the state-of-the-art CNN based method (Ballé) in terms of PSNR. While measured by MS-SSIM, the proposed method outperforms all the traditional methods. Although the proposed method consumes 9.41% higher rate in terms of BDBR [24] on PSNR compared with Ballé's, the proposed method can save substantial time to encode or decode an image as shown in Table I.

The executing time results of all the comparative methods shown in Table I are tested on a computer with 8 logical Intel(R) Core(TM) i7-9700K CPU @ 3.60GHz and a GeForce GTX 1080 Ti GPU. The quality target is set as 40dB in terms of PSNR. To simulate the transmitting process in practical Internet environment, the bandwidth is assumed as 10Mbps. The results include the time of encoding, transmitting, decoding the preview image (if the method supports the progressive coding) and decoding the full-quality image. Here we define the user waiting time as the time of loading the preview image (for progressive coding) or the full-quality image (for non-progressive coding).

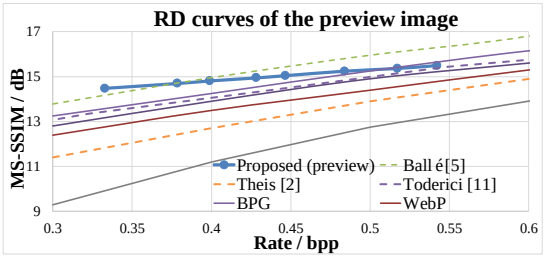

Fig. 5: RD curves of the preview image. Note that, among these methods, Webp, BPG, Theis' [2] and Ballé's [5] methods don't support progressive coding and they are tested using the normal coding mode at low rates.

As we can see, the methods doesn't support progressive coding can't load the image until the full-size file ($\boldsymbol{B}_{ful}$) is totally transmitted and the full-quality image $\hat{\boldsymbol{I}}_{ful}$ is decoded. These methods have long user waiting time. In contrast, the proposed method can quickly load the preview image $\hat{\boldsymbol{I}}_{pre}$ from a part of the file $\boldsymbol{B}_{bas}$ with real-time efficiency, leading to negligible user waiting time (only 33.0 ms). Moreover, even if measured by the loading time of the full-quality image, the proposed method outperforms all the comparative methods except JPEG, which has much lower coding performance than the proposed method. Compared with Ballé's method, the proposed method can save 40.46% or 30.60% to encode or load an image (sum of transmitting and decoding time) respectively with only 9.41% BDBR performance decline, and the proposed method can decode a preview image with only 25.50% time. As the relationship between the rate and the user waiting time shown in Fig. 1, the proposed method achieves a good balance between the coding performance and the executing efficiency for practical application.

We show the quality of the preview image by RD curves as shown in Fig. 5. The eight low-rate points of the preview image compression are generates together with those eight full-quality compression results shown in Fig. 4. As we can

Table I: Time testing averaged on Kodak in terms of encoding, transmitting and decoding time. We assume the bandwidth is 10Mbps and the transparency quality threshold is 40 dB in terms of PSNR. Note that the executable program of Toderici's method can't generate full-quality image with higher 40 dB PSNR.

Methods	Encoder end		Decoder end (accumulated time is shown in the right column)								User waiting time
	Encode	File size	Receive $\boldsymbol{B}_{pre}$		Decode $\hat{\boldsymbol{I}}_{pre}$		Receive $\boldsymbol{B}_{ful}$		Decode $\hat{\boldsymbol{I}}_{ful}$		
JPEG	**26.0**	149.7Kb	18.7	**18.7**	5.8	**24.5**	116.9	116.9	9.2	126.1	**24.5**
JPEG2000	57.0	119.0Kb	18.8	18.7	57.9	76.6	92.9	92.9	58.6	151.6	76.6
WebP	119.3	104.1Kb	-	-	-	-	81.3	81.3	156.2	237.5	237.5
BPG	281.7	83.1Kb	-	-	-	-	64.9	64.9	145.9	210.9	210.9
Balle [5]	78.6	75.4Kb	-	-	-	-	58.9	**58.9**	70.5	129.4	129.4
Todrici [11]	899.1	-	20.3	20.3	242.8	263.1	-	-	-	-	263.1
Proposed	46.8	84.0Kb	18.8	18.8	14.2	33.0	65.6	65.6	24.2	**89.8**	33.0

see, the proposed method can compress the preview image with high coding performance that is competitive to Ballé's method or BPG and even better than them at rates lower 0.35bpp. Note that neither Ballé's method nor BPG supports progressive coding and they are ran in the normal mode at low rates but with full computational complexities.

IV. Conclusion

This paper proposes a novel CNN based progressive image compression framework to solve the conflict between the high quality requirements and the long loading delay with high coding performance and executing efficiency. The proposed framework is composed of a unified encoder network and two progressive decoder networks, with which the image can be coded into a preview image and a full-quality image. All the networks are jointly optimized for the minimized rate distortion losses of the preview and the full quality images. Experiments results show the proposed method achieves state-of-the-art coding performance at high rates, and the executing efficiency is close to JPEG (but with significantly higher coding performance) and better than all other comparative methods.

Acknowledgment

This work was supported in part by the Natural Science Foundation of Shanghai under Grant 18ZR1418100, in part by the National Natural Science Foundation of China under Grant 61771306 and Grant 61527804, and in part by the the Shanghai Key Laboratory of Digital Media Processing and Transmissions under Grant STCSM18DZ2270700.

References

[1] J. Mitchell, "Digital compression and coding of continuous-tone still images: Requirements and guidelines," *ITU-T Recommendation T*, vol. 81, 1992.

[2] L. Theis, W. Shi, A. Cunningham, and F. Huszár, "Lossy image compression with compressive autoencoders," *International Conference on Learning Representations*, 2017.

[3] E. Agustsson, F. Mentzer, M. Tschannen, L. Cavigelli, R. Timofte, L. Benini, and L. V. Gool, "Soft-to-hard vector quantization for end-to-end learning compressible representations," in *Advances in Neural Information Processing Systems*, 2017, pp. 1141–1151.

[4] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," *International Conference on Learning Representations*, 2017.

[5] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," *International Conference on Learning Representations*, 2018.

[6] D. Minnen, J. Ballé, and G. D. Toderici, "Joint autoregressive and hierarchical priors for learned image compression," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 793–10 802.

[7] L. Zhou, C. Cai, Y. Gao, S. Su, and J. Wu, "Variational autoencoder for low bit-rate image compression," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 2617–2620.

[8] C. Cai, L. Chen, X. Zhang, and Z. Gao, "Efficient variable rate image compression with multi-scale decomposition network," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1, 2018.

[9] N. Johnston, D. Vincent, D. Minnen, M. Covell, S. Singh, T. Chinen, S. J. Hwang, J. Shor, and G. Toderici, "Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks," *structure*, vol. 10, p. 23.

[10] M. Li, W. Zuo, S. Gu, D. Zhao, and D. Zhang, "Learning convolutional networks for content-weighted image compression," *Computer Vision and Pattern Recognition (CVPR), 2018 IEEE Conference on*, 2018.

[11] G. Toderici, D. Vincent, N. Johnston, S. J. Hwang, D. Minnen, J. Shor, and M. Covell, "Full resolution image compression with recurrent neural networks," in *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE, 2017, pp. 5435–5443.

[12] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimization of nonlinear transform codes for perceptual quality," in *Picture Coding Symposium (PCS), 2016*. IEEE, 2016, pp. 1–5.

[13] G. Lu, W. Ouyang, D. Xu, X. Zhang, Z. Gao, and M.-T. Sun, "Deep kalman filtering network for video compression artifact reduction," in *The European Conference on Computer Vision (ECCV)*, September 2018.

[14] G. Lu, W. Ouyang, D. Xu, X. Zhang, C. Cai, and Z. Gao, "Dvc: An end-to-end deep video compression framework," *arXiv preprint arXiv:1812.00101*, 2018.

[15] C. Cai, G. Lu, Q. Hu, L. Chen, and Z. Gao, "Efficient learning based sub-pixel image compression," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

[16] F. Bellard, "Bpg image format (http://bellard. org/bpg/)," 2017.

[17] J. Ballé, V. Laparra, and E. P. Simoncelli, "Density modeling of images using a generalized normalization transformation," *International Conference on Learning Representations*, 2016.

[18] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multiscale structural similarity for image quality assessment," in *Signals, Systems and Computers, 2004. Conference Record of the Thirty-Seventh Asilomar Conference on*, vol. 2. IEEE, 2003, pp. 1398–1402.

[19] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," California Univ San Diego La Jolla Inst for Cognitive Science, Tech. Rep., 1985.

[20] Google, "Webp: Compression techniques," *URL http://developers.google.com/speed/webp/docs/compression.*, 2018.

[21] M. Rabbani, "Jpeg2000: Image compression fundamentals, standards and practice," *Journal of Electronic Imaging*, vol. 11, no. 2, p. 286, 2002.

[22] G. K. Wallace, "The jpeg still picture compression standard," *IEEE transactions on consumer electronics*, vol. 38, no. 1, pp. xviii–xxxiv, 1992.

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

[24] G. Bjontegaard, "Calculation of average psnr differences between rd-curves," in *ITU-T Q. 6/SG16 VCEG, 15th Meeting, Austin, Texas, USA, April, 2001*, 2001.