Binary Probability Model for Learning Based Image Compression

Théo Ladune, Pierrick Philippe, Wassim Hamidouche, Lu Zhang, Olivier Deforges

BINARY PROBABILITY MODEL FOR LEARNING BASED IMAGE COMPRESSION

Théo LADUNE[†], Pierrick PHILIPPE[*], Wassim HAMIDOUCHE[†], Lu ZHANG[†], Olivier DÉFORGES[†]*

[*]Orange Labs, 4 rue du Clos Courtel, 35512, Cesson-Sévigné, France
firstname.lastname@orange.com
[†]Univ. Rennes, INSA Rennes, CNRS, IETR – UMR 6164, Rennes, France
firstname.lastname@insa-rennes.fr

ABSTRACT

In this paper, we propose to enhance learned image compression systems with a richer probability model for the latent variables. Previous works model the latents with a Gaussian or a Laplace distribution. Inspired by binary arithmetic coding, we propose to signal the latents with three binary values and one integer, with different probability models.

A relaxation method is designed to perform gradient-based training. The richer probability model results in a better entropy coding leading to lower rate. Experiments under the Challenge on Learned Image Compression (CLIC) test conditions demonstrate that this method achieves 18 % rate saving compared to Gaussian or Laplace models.

***Index Terms*—** Image Coding, Autoencoder, Entropy Coding, Convolutional Neural Network

1. INTRODUCTION

Data compression can be summarized in three main steps. First, the input signal is encoded into more compact variables called latents. Then, the latents are transmitted with a coding method achieving a rate near to the Shannon entropy. Lastly, the input signal is decoded from the latents. As a real number has an infinite information quantity (*i.e.* an infinite number of bits), lossy coding methods only work with finite set of values. To address this issue, latents are quantized, introducing distortion on both the latents and the reconstructed signal.

Lossy image compression can thus be expressed as an optimization problem: jointly minimizing the distortion and the rate (*i.e.* information in the latents). Traditional coding approaches such as JPEG or BPG (HEVC-based image compression) [1, 2] typically solve this problem using linear predictions and transforms. Deep neural networks can learn complex non-linear functions, making them well suited to reach better optimum and coding efficiency. However, the discrete nature of the data sent from the encoder to the decoder makes the objective function non-differentiable and prevents optimizing end-to-end systems with gradient-based methods.

In [3], authors suggest to replace quantization with additive noise and propose an interpolation of the rate function. A different quantization approximation is presented in [4]. These works show promising results, outperforming the JPEG standard.

Entropy coding requires an estimate of the latents probability density function (PDF). Whereas previous works use a fixed-PDF model, Ballé *et al.* introduce hyperpriors in [5], consisting in side-information conditioning each latent PDF. This more accurate probability model brings important performance gains. Minnen *et al.* and Lee *et al.* [6, 7] add an autoregressive model (ARM) to infer PDF parameters from previously sent values. However, such systems lead to a prohibitive decoding time due to the sequential nature of the ARM which is not suited for GPU processing.

In 2019, the Challenge on Learned Image Compression (CLIC) [8] was held at the Conference on Computer Vision and Pattern Recognition (CVPR), providing a common evaluation framework to the learned image compression community. Proposed end-to-end systems [9, 10] composed of a hyperprior and an ARM outperformed BPG [2].

Improvements of the latents probability model are the main reason behind the successive performance gains. In this paper, we propose a more accurate estimate of the latents PDF widely inspired by the HEVC binarization process [11]. Based upon Minnen's work [6], we present a new relaxation method for a discrete rate function. This allows to leverage the richer probability model providing either better performance with the same complexity or similar performance with a lightweight coding system.

2. PROPOSED METHOD

2.1. Framework description

The work carried out in this paper is based upon Ballé and Minnen's work [3, 5, 6]. Their framework for training end-to-end lossy compression system is explained in this section. The architecture is the one described in [6]. Fig. 1 illustrates the coding scheme which can be summarized as:

1. Encoding the input image $\mathbf{x}$ into latents $\mathbf{y} = g_a(\mathbf{x}; \boldsymbol{\theta}_e)$;
2. Encoding the hyperprior $\mathbf{z} = h_a(\mathbf{y}; \boldsymbol{\theta}_{he})$;

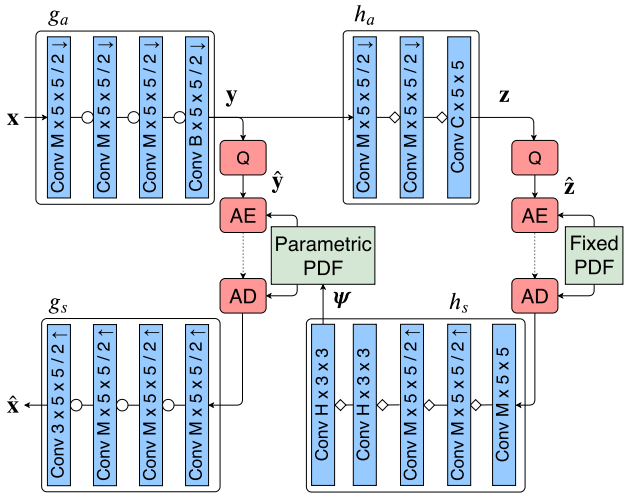

Fig. 1: Network architecture. Rounded arrows denote GDN [3] and squared arrows LeakyReLU. Convolution parameters are: filters number × kernel height × width / stride. Upscaling convolutions are transposed convolutions.

3. Quantizing $\hat{\mathbf{z}} = Q(\mathbf{z})$, $\hat{\mathbf{y}} = Q(\mathbf{y})$ with a unitary uniform scalar quantizer;
4. Lossless arithmetic encoding (AE) and decoding (AD);
5. Decoding PDF parameters $\boldsymbol{\psi} = h_s(\hat{\mathbf{z}}; \boldsymbol{\theta}_{hd})$;
6. Decoding $\hat{\mathbf{y}}$ to reconstruct the input image $\hat{\mathbf{x}} = g_s(\hat{\mathbf{y}}; \boldsymbol{\theta}_d)$.

The set of neural network parameters $\{\boldsymbol{\theta}_e, \boldsymbol{\theta}_d, \boldsymbol{\theta}_{he}, \boldsymbol{\theta}_{hd}\}$ is learnt by minimizing a rate-distortion trade-off

$$\mathcal{L}(\lambda) = D(\mathbf{x}, \hat{\mathbf{x}}) + \lambda \left(R(\hat{\mathbf{y}}) + R(\hat{\mathbf{z}}) \right).$$

In this work, the distortion is computed through the mean-squared error $D(\mathbf{x},\ \hat{\mathbf{x}}) = \mathbb{E}_{\mathbf{x} \sim p_{\mathbf{x}}} \left[||\mathbf{x} - \hat{\mathbf{x}}||^2 \right]$.

Latents $\hat{\mathbf{y}}$ and the hyperprior $\hat{\mathbf{z}}$ are encoded with arithmetic coding, a lossless coding method achieving a rate near to Shannon entropy

$$R(\hat{\mathbf{y}}) = \mathbb{E}_{\hat{\mathbf{y}} \sim m}[L(\hat{\mathbf{y}}; \mathbb{P}_{\hat{\mathbf{y}}})] = \mathbb{E}_{\hat{\mathbf{y}} \sim m}[-\log_2 \mathbb{P}_{\hat{\mathbf{y}}}(\hat{\mathbf{y}})],$$

where m denotes the distribution of latents (which is unknown) and L is the code length computed thanks to the probability model $P_{\hat{\mathbf{y}}}$. This can be re-written as [7]:

$$R(\hat{\mathbf{y}}) = H(m) + D_{KL}(m \,||\, \mathbb{P}_{\hat{\mathbf{y}}}),$$

where D_{KL} denotes the Kullback-Leibler divergence. Thus, minimizing the rate implies to jointly lower the entropy $H(m)$ of $\hat{\mathbf{y}}$ and properly match the distribution m with the probability model $\mathbb{P}_{\hat{\mathbf{y}}}$. This also holds for rate of $\hat{\mathbf{z}}$.

Training neural networks relies on gradient-based algorithms, requiring all operations to be differentiable. Because quantization derivative is null almost everywhere, it is modeled as an additive uniform noise during training [3]

$$\tilde{\mathbf{y}} = \mathbf{y} + \mathbf{u} \Rightarrow p_{\tilde{\mathbf{y}}} = p_{\mathbf{y}} * p_{\mathbf{u}},\ \mathbf{u} \sim \mathcal{U}(-\tfrac{1}{2},\ \tfrac{1}{2}),$$

where p denotes probability distribution. Continuous interpolation $\tilde{L}(\tilde{\mathbf{y}}; p_{\tilde{\mathbf{y}}}) = -\log_2 p_{\tilde{\mathbf{y}}}(\tilde{\mathbf{y}})$ of the code length function is used as a proxy to optimize discrete $L(\hat{\mathbf{y}}; P_{\hat{\mathbf{y}}})$. The same goes for $\hat{\mathbf{z}}$ and the loss function becomes

$$\mathcal{L}(\lambda) = \mathbb{E}_{\mathbf{x} \sim p_{\mathbf{x}}} \left[||\mathbf{x} - \hat{\mathbf{x}}||^2 + \lambda(\tilde{L}(\tilde{\mathbf{y}}; p_{\tilde{\mathbf{y}}}) + \tilde{L}(\tilde{\mathbf{z}}; p_{\tilde{\mathbf{z}}}) \right]. \quad (1)$$

The hyperprior distribution $p_{\tilde{\mathbf{z}}}$ is estimated through a fixed model described in [5]. Each latent y_i is coded independently and their distribution $p_{y_i} \sim \mathcal{N}(\mu_i, \sigma_i)$ is decoded from the hyperprior

$$\begin{aligned} \tilde{L}(\tilde{\mathbf{y}}; p_{\tilde{\mathbf{y}}}) &= \sum_i \tilde{L}(\tilde{y}_i, p_{\tilde{y}_i}) = \sum_i -\log_2 \left(p_{y_i} * p_u \right)(\tilde{y}_i) \\ &= \sum_i -\log_2 \int_{\tilde{y}_i - \frac{1}{2}}^{\tilde{y}_i + \frac{1}{2}} \mathcal{N}(u;\ \mu_i, \sigma_i)\, \mathrm{d}u. \end{aligned} \quad (2)$$

In this paper, we enhance the probability model p_{y_i} in order to improve the entropy coding efficiency. As in traditional video coding, latents are transmitted in a binary version, allowing a more accurate model p_{y_i}.

2.2. Binary probability model

For the sake of clarity, latents index is omitted *i.e.* y stands for any y_i. The purpose of this work is to relax assumptions on p_y. To do so, each latent is represented with three binary values and one integer with separate probability model. First, the expectation μ is decoded from the hyperprior and used to center y before quantization: $\hat{y} = Q(y - \mu)$. Each $\hat{y}$ is then signaled as described in Table 1.

$\hat{y}$	Elements transmitted				Code length L_{bin}
	G_0	G_1	S	E	
0	0				L_{G_0}
± 1	1	0	± 1		$L_{G_0} + L_{G_1} + L_S$
$\pm k$	1	1	± 1	k	$L_{G_0} + L_{G_1} + L_S + L_E$

Table 1: G_0 (respectively G_1) stands for greater than zero (respectively one), S for sign and E for explicit.

Flags G_0 and G_1 are transmitted using an entropy coding method, their code length is estimated as

$$L_{G_X} = \begin{cases} -\log_2 P_{G_X} & \text{if } G_X = 1, \\ -\log_2 (1 - P_{G_X}) & \text{otherwise} \end{cases} \quad X = \{0,\ 1\}.$$

Probabilities P_{G_0} and P_{G_1} are decoded from the hyperprior $\hat{\mathbf{z}}$. The sign flag is assumed equiprobable costing $L_S = 1$ bit. A latent $|\hat{y}| \geq 2$ is explicitly transmitted with a code length estimated as

$$L_E(k) = -\log_2 P_{\hat{y}}(|\hat{y}| = k \,\big|\, |\hat{y}| > 1). \quad (3)$$

Here, p_y is modelled as a centered Laplace distribution with σ decoded from the hyperprior. Equation (3) becomes

$$L_E(k) = -\log_2 \left(\frac{2 \int_{k-0.5}^{k+0.5} \mathcal{L}(u; 0, \sigma)\, \mathrm{d}u}{1 - \int_{-1.5}^{1.5} \mathcal{L}(u; 0, \sigma)\, \mathrm{d}u} \right). \quad (4)$$

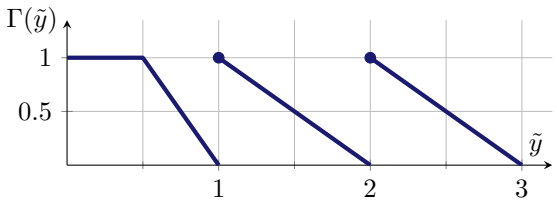

Fig. 2: The weighting function Γ.

The total code length L_{bin} is obtained by adding up all transmitted elements (*cf.* Table 1). All $\hat{y} \in \{-1,\ 0,\ 1\}$ are no longer constrained to a pre-determined distribution as $P_{\hat{y}}$ can represent any symmetrical probability distribution in this interval. The entropy coding of each latent y requires the set $\{\mu, \sigma, P_{G_0}, P_{G_1}\}$. Hence, the decoded hyperprior $\boldsymbol{\psi}$ has four features per $\hat{y}$: in Fig. 1 $H = 4B$.

2.3. Relaxed rate

The previous section proposes a richer representation of $P_{\hat{y}}$. During training, discrete $\hat{\mathbf{y}}$ is replaced by a continuous $\tilde{\mathbf{y}}$, requiring the interpolation of the code length function $\tilde{L}$. As no hypothesis is made on p_y, eq. (2) can not be used directly. A new interpolation $\tilde{L}_{bin}$ is introduced as a weighted sum of the two nearest integer rates:

$$\tilde{L}_{bin}(\tilde{y}) = \Gamma(|\tilde{y}|)L_{bin}(\lfloor\tilde{y}\rfloor) + (1 - \Gamma(|\tilde{y}|))\, L_{bin}(\lfloor\tilde{y}\rfloor + 1),$$

where $\lfloor\cdot\rfloor$ denotes the floor function. $\Gamma(\tilde{y})$ is a weighting function defined with linear segments and depicted in Fig. 2. The main design constraint on the weighting function Γ is to ensure that $\tilde{L}_{bin}(k) = L_{bin}(k)$ for all integers k to make training and inference metrics coherent. Because sending $\hat{y} = 0$ requires only one element (G_0), the optimization process results in zeros being the most present value. The flat zone in $[0, \frac{1}{2}]$ is used to make the optimization focus more on the cost of zeros. In $[1, +\infty]$ interval, Γ is a simple linear weighting based on the distance to the nearest integer. With the relaxed rate, the loss function becomes:

$$\mathcal{L}(\lambda) = \mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}}[||\mathbf{x} - \hat{\mathbf{x}}||^2 - \lambda(\tilde{L}_{bin}(\tilde{\mathbf{y}}) + \tilde{L}(\tilde{\mathbf{z}}; p_{\tilde{\mathbf{z}}}))].$$

3. EXPERIMENTAL RESULTS

3.1. Performance on CLIC low-rate task

The proposed method is evaluated on the CLIC 2019 low-rate task [8]. The objective is to achieve the highest PSNR at 0.15 bit per pixel (bpp). For all experiments, the training set is constructed by concatenating the CLIC and DIV2K [12] datasets. The 3 000 pictures of these datasets are transformed into non-overlapping 256×256 crops. Minibatches of size 8 and Adam algorithm with a learning rate of 10^{-4} are used. The training lasts 80 epochs and the learning rate is divided by 5 at the 50$^{\text{th}}$ and 70$^{\text{th}}$ epoch.

The network described in Fig. 1 is used to evaluate three probability models: Gaussian, Laplace and binary. For all experiments, $B = 76$ $\hat{\mathbf{y}}$ features and $C = 24$ $\hat{\mathbf{z}}$ features are transmitted. Transforms g_a, g_s and h_a always have the same complexity. The transform h_s is slightly modified due to the number of features (denoted as H in Fig. 1) needed to parameterize latents distribution ($H = 2B$ for Gaussian and Laplace, $H = 4B$ for binary model). Hence, different performance levels are entirely explained by the probability model. The models are evaluated with lightweight ($M = 64$) and standard ($M = 192$) configurations.

The rate is estimated by the latents entropy. Performance at 0.15 bpp is obtained by training systems with a λ setting a working point close to the target rate. During inference, the quantization step can be slightly deviated from 1 to plot rate distortion curve around the training point. This enables to accurately estimate the rate at 0.15 bpp and to compute BD rates [13] by comparing RD curves in [0.13, 0.17] bpp interval. BD rate represents the rate difference necessary to obtain the same PSNR quality between two systems.

Figure 4 and Table 3 sum up results on CLIC 2019 validation and test sets, composed of 102 and 330 various resolution images . Gaussian systems are re-implementations of Minnen et al. [6] without the autoregressive component and are used as a baseline. Laplacian is added as [14] argues that it slightly improves performances. BPG is also added as it is the image version of HEVC, the state-of-the-art video coding standard.

The proposed method shows significant rate savings in all configurations, up to 18.3 %. This proves the benefits of a richer PDF model to perform a more efficient entropy coding. Binary probability model brings 9.1 % rate saving for standard systems, achieving results competitive with BPG. Performance improvements are greater with lightweight systems. It may be because they have less powerful transforms g_a and g_s. Indeed, relaxing the constraints $p_{\mathbf{y}}$ makes the system focus more on creating useful latents instead of matching a given PDF. This holds for standard systems to a lesser extent. Finally, it is worth noting that the binary model lightweight system can reach the performance of the standard Gaussian system with 10 times less parameters.

3.2. Illustration

Figure 5 depicts the processing of an image by the binary model system. On the left side, feature map $\hat{\mathbf{y}}_{65}$ is the costliest feature map (around 7 % of the rate). Many pixels are greater than one, resulting in high probabilities for $\boldsymbol{P_{G_0}}$ and $\boldsymbol{P_{G_1}}$. As most of the values have important dynamic and need explicit sending, the scale parameter $\boldsymbol{\sigma}$ takes a wide range of values. On the right side, feature map $\hat{\mathbf{y}}_{51}$ is very sparse and consists mostly in details, representing only 2 % of the rate. Entirely null areas, as the sky, are well captured by the hyperprior, with a very low probability of being greater than zero. This allows to code them with fewer bits.

Systems	M	Validation PSNR [dB]	Validation BD rate [%]	Test PSNR [dB]	Test BD rate [%]
JPEG	/	26.31	/	25.10	/
BPG		30.84		29.60	
Gaussian	64	30.10	Ref.	28.87	Ref.
Laplacian		30.22	-5.9	28.99	-7.5
Binary		**30.48**	**-14.4**	**29.26**	**-18.3**
Gaussian	192	30.56	Ref.	29.31	Ref.
Laplacian		30.51	2.1	29.26	3.1
Binary		**30.68**	**-7.5**	**29.49**	**-9.1**

Fig. 3: Latents probability models performances on CLIC validation and test sets. PSNR are given at 0.15 bpp. BD rates are computed with the Gaussian system as reference.

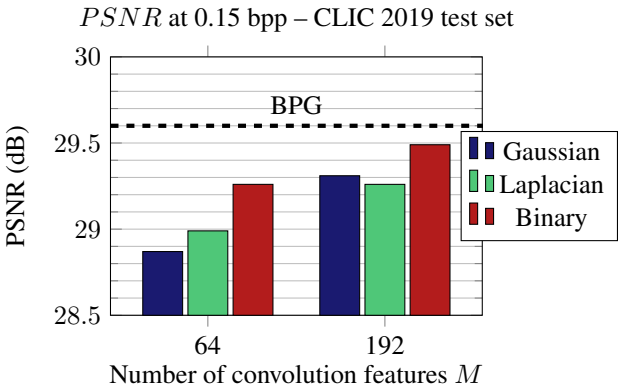

Fig. 4: Latents probability models performances.

4. CONCLUSION

This paper proposes a richer latents probability model based on binary values and a learning process adapted for gradient-based training. Experimental results demonstrates that this method achieves important gains compared to usual parametric models such as Gaussian and Laplace distributions. Under the CLIC test conditions, the binary probability model leads to a rate saving up to 18 % for the same reconstruction quality. In future work, the binary model could be made even more generalist with additional flags (G_2, G_3 *etc.*). This would reduce latents explicit sending frequency and increase the coding performance. The autoregressive component could be used simultaneously with the proposed binary model to study their interactions.

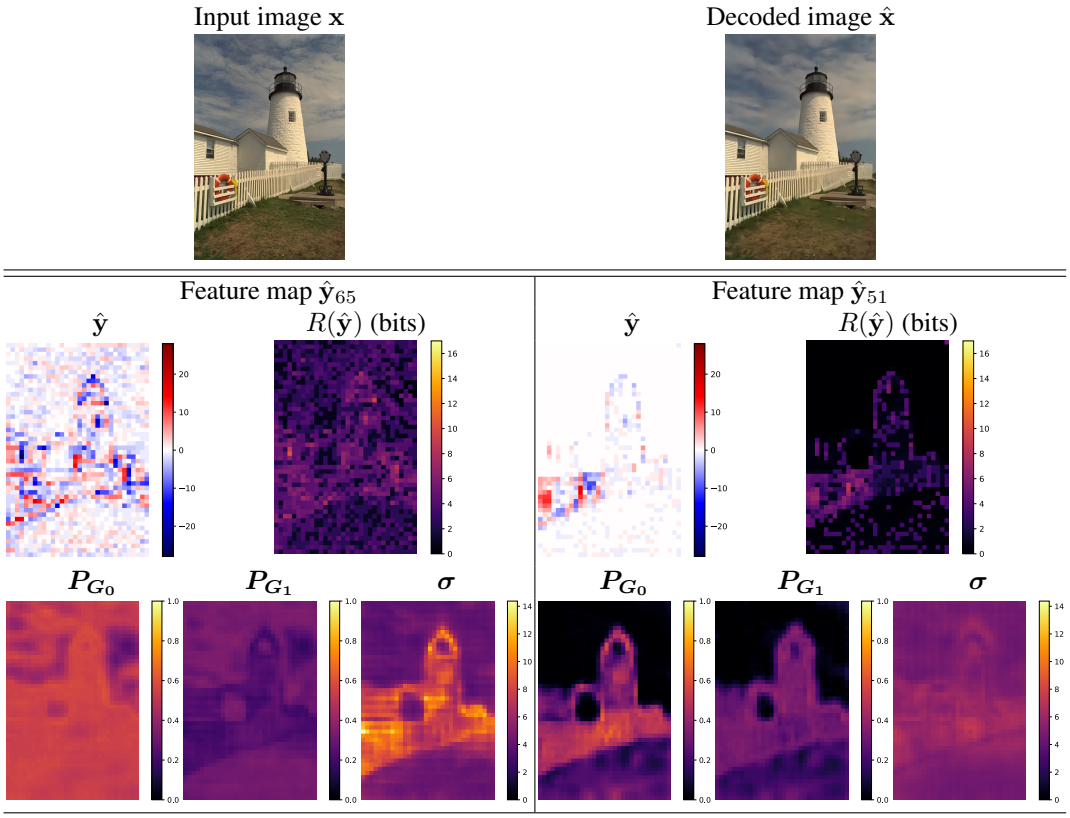

Fig. 5: Top: Original and compressed image. Middle: two $\hat{\mathbf{y}}$ and their corresponding rate. Bottom: $\boldsymbol{P_{G_0}}$ (respectively $\boldsymbol{P_{G_1}}$) is the probability for a pixel to be greater than 0 (respectively 1). $\boldsymbol{\sigma}$ is the scale parameter used for explicit latents sending.

5. REFERENCES

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

[2] Fabrice Bellard, "https://bellard.org/bpg/," 2014.

[3] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli, "End-to-end optimized image compression," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. 2017, OpenReview.net.

[4] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár, "Lossy image compression with compressive autoencoders," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. 2017, OpenReview.net.

[5] 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 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. 2018, OpenReview.net.

[6] David Minnen, Johannes Ballé, and George Toderici, "Joint autoregressive and hierarchical priors for learned image compression," in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, Eds., 2018, pp. 10794–10803.

[7] 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, New Orleans, LA, USA, May 6-9, 2019*. 2019, OpenReview.net.

[8] Workshop and Challenge on Learned Image Compression, "https://www.compression.cc/," June 2019.

[9] Lei Zhou, Zhenhong Sun, Xiangji Wu, and Junmin Wu, "End-to-end optimized image compression with attention mechanism," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[10] Sihan Wen, "Variational autoencoder based image compression with pyramidal features and context entropy model," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[11] Gary J. Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand, "Overview of the high efficiency video coding (hevc) standard," *IEEE Trans. Cir. and Sys. for Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[12] Eirikur Agustsson and Radu Timofte, "Ntire 2017 challenge on single image super-resolution: Dataset and study," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.

[13] Gisle Bjontegaard, "Calculation of average psnr differences between rd-curves," in *ITU-T Q.6/16, Doc. VCEG-M33*, March 2001.

[14] Lei Zhou, Chunlei Cai, Yue Gao, Sanbao Su, and Junmin Wu, "Variational autoencoder for low bit-rate image compression," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2018.