Rate-Constrained Learning-based Image Compression

Nilson D. Guerin Jr.[a,*], Renam Castro da Silva[c], Matheus C. de Oliveira[b], Henrique C. Jung[a], Luiz Gustavo R. Martins[a], Eduardo Peixoto[b], Bruno Macchiavello[a], Edson M. Hung[b], Vanessa Testoni[c], Pedro Garcia Freitas[c]

[a]*Department of Computer Science, University of Brasilia, Brazil*
[b]*Electrical Engineering Department, University of Brasilia, Brazil*
[c]*Samsung Eletrônica da Amazonia Ltda, Brazil*

Abstract

Rate control is a desirable feature, sometimes a requirement, for several applications in still image coding. Usually, the objective is to achieve rate control for every input data with minimal impact on rate-distortion performance. However, this task can be quite challenging. Learning-based image compression is a new paradigm that needs to be competitive with conventional image coding techniques. A learning-based lossy codec may require several trained models for different quality requirements. Therefore, a coding tool providing the ability to achieve a specific rate can be a deterministic factor to apply such models in practical application scenarios. Hence, in this work, we present a non-constrained solution to solve the constrained problem of training a learning-based image codec for a specific bitrate. The proposed solution requires a modified loss function for autoencoder optimization. This modification allows controlling the deviation from the specified target rate. Experiments performed in Kodak and JPEG AI datasets show that autoencoders trained with the proposed loss function can achieve rate constrained encoding with negligible losses in terms of SSIM and MS-SSIM.

Keywords: Image Coding, Neural Networks, Rate-distortion, Rate Control, Learning-based Compression

1. Introduction

Image compression is an essential task in digital communication, broadcasting, and storage. Traditionally, it is based on techniques such as transform coding, prediction techniques, and scalar quantization. These techniques have been employed in several standards. For instance, JPEG-1[1], a DCT-based codec, has prevailed in the field of practical lossy image compression since its introduction in 1992. Its successor, JPEG 2000 [2], has been used in digital cinema, medical applications, and other professional markets.

Newer standards with different objectives have been proposed in recent years. The JPEG XR [3] standard has targeted better compression than JPEG-1 with limited complexity overhead. However, these standards have performance limitations for HDR images. To overcome this limitation, the JPEG committee released the JPEG XT [4], a backward-compatible standard supporting compression of HDR images. The committee also launched the JPEG XL [5] standardization initiative, aiming at developing the next-generation image coding system with features amenable for web distribution and support for handling HD and UHD images, wide color gamut, and high bit depth images, while being backward compatible with JPEG-1. Recently, a new coding paradigm based on DNN was introduced and has been steadily improved from barely reaching JPEG-1 performance to methods that can surpass HEVC intra-coding [6, 7, 8, 9].

Toderici et al. [10] proposed one of the first architectures for encoding images using DNN based on LSTM lay-

*Corresponding author
Email address: nguerinjr@gmail.com (Nilson D. Guerin Jr.)

ers in combination with fully-connected and convolutional residual layers. In this architecture, the residuals at each iteration are progressively encoded to achieve higher bitrates and better quality, which presents a special characteristic of being scalable and not requiring multiple training to reach the desired bitrate. Minnen et al. [11] improved this model by including an intra-prediction stage before encoding. Later, Jung et al. [12] enhanced Toderici's work by adding a block-based multi-mode intra-prediction. Although flexible in terms of rate, this proposal has significant computational complexity and limited compression performance.

Ballé et al. [13, 14] proposed one of the most relevant methods in terms of rate-distortion performance with an end-to-end autoencoder. This model was improved later [7] by introducing a hyperprior to capture spatial dependencies and get rid of statistical redundancy in the entropy model for a further rate reduction. Several works [15, 16, 17] have been done to improve upon Ballé's model.

Despite the interest and advances in the field of learning-based image compression, these methods have not been widely employed in practical applications. Some reasons include diffidence in metrics, lead time required to establish and embody a novel standard, and demanding computational requirements for seamless deployment. Another problem to be surpassed is the lack of techniques for the definition of bit allocation and rate control for DNN-based codecs. Toderici's proposal [10] enables the user to define a target rate, however, to attain the target rate, the encoding is performed in a sequence of steps which is increased as the target rate is set to a higher value, ending up being a complex solution to be adopted in practical applications. Alternatively, Ballé's model requires several networks to be trained to provide a set of rate-distortion (RD) operation points. However, as one would expect, each trained model outputs a considerable deal of variation in terms of rate and distortion for different images.

In this work, we propose a modification of the loss function to introduce rate control in neural codecs. The idea is to make each trained model operate on a target rate with a more strict rate variation for different images. We also propose a set of heuristics to determine the new parameters introduced by our modification. Commonly, hyperparameters are determined empirically and this may require several training cycles. The heuristics presented in this work, avoid this empirical search for the new hyperparameters that are introduced. The results are validated using the Kodak dataset [18] and JPEG AI dataset [19].

2. Literature Review

A variational autoencoder (VAE) is a neural network that implements an approximate model of the Bayesian *a posteriori* inference. Proposed in [20], it has been widely used to model the generation process of data. The density approximation capability of these architectures has also made them useful for data coding. The first seminal works in the field of image coding [21, 14] proposed the implementation of a VAE through optimization of a Lagrangian rate-distortion loss function. In this context, the *a posteriori* approximation is used to model the density of latent representation. The VAE architecture for image coding was later modified with the inclusion of a hyperprior component [7] that employs an auxiliary VAE that outputs the parameters of the entropy model of the main latent. This hyperprior architecture was further extended with the adoption of an auto-regressive component [8]. This component is responsible for the backward context analysis of the code and refines the parameterization of the latent distribution, along with the hyperprior component.

Previous works [14, 7, 8] establish the main paradigm for image compression using VAE, as they originally surpassed other approaches. Since then, there has been an intense interest in the area of image compression using learning-based approaches. Many approaches were proposed for improving and exploring these architectures in

different ways.

In [22], the authors propose to replace the global entropy model with a set of different generated entropy models which are chosen with respect to the image. In [23], conditions of independence of the different components of the latent representation were adopted to alleviate the problem of non-parallelism of the auto-regressive component, making it more feasible. A multi-scale approach of the auto-regressive component was also studied to improve the performance of the context modeling [24].

The work [21] investigated more efficient ways to train multiple neural models resorting to fine-tuning of the hyperparameters. Whereas [25] adopted a channel-wise approach to increase the parallelism of the predictions in auto-regressive context models. The auto-regressive component is replaced by a hierarchy of hyperpriors in [26], providing similar efficiency and high parallelism. Prune strategies are investigated in [27] to obtain models with similar efficiency and reduced number of network parameters.

Different convolution operations have also been explored to improve the capacity of the neural models. Introduced in [28], the so-called octave-convolution is modified and adopted in [29] to replace downsampling and upsampling operations with learned convolutions. [30] proposed an extension of these operations including a learned gain factor λ, calling them as modulated octave convolutions. Works approaching post-processing techniques [31], generative compression [9, 32], wavelet-inspired approaches [33] can also be seen in the literature.

One important issue is that all previously mentioned VAE approaches require one model for each point in the rate-distortion curve, which is variable through different images. This represents a high computational effort during training and a high storage/transmission cost of the different models. Therefore, another line of research explored the design of a single model to achieve multiple rates. One of the first approaches [34], proposes to explicitly learn the quantization step size for each feature map of a single model. At test time, different rates are achieved by scaling the learned quantization step sizes. Additionally, it also experimented with varying the quantization step size of a single model produced by the baseline. Experiments revealed the proposed techniques were able to produce performance comparable to the approach of multiple models.

Inspired by [35, 36], the authors of [37] propose an approach that uses the Lagrangian multiplier of the rate-distortion function as a conditioning parameter, achieving multiple rates for each image through the use of this parameter. In a subsequent work [38], a multiscale representation of the latent is adopted, where each composed representation yields different reconstructions with different qualities. A gained variational autoencoder is proposed in [39] where a gain unity can be adjusted to modify the rate. A more sophisticated approach was also proposed where a Bayesian arithmetic coder replaced the rigid uncertainty region of the original encoder achieving different rates with this modification [40].

More recently, the work [41] explored approaches to provide variable rate compression with a single model and analyzed the power representation of nonlinear transforms. The authors [41] presented an entropy-constrained vector quantization formulation that may be used as an empirical lower bound for nonlinear transform coding. Moreover, targeting to traverse the rate-distortion trade-off, they exploit several ways to embed the Lagrangian multiplier as a conditioning factor of the transform inner operations and entropy model, extending [37] .

Other studies have focused on bit allocation and attention modules. Non-local residual attention modules [42] are used in [43] to give the network ability to allocate more bits in regions relevant to the distortion measure. These modules are also applied to other works [44, 45] to improve the discovery of local and non-local correlations. Authors in [46] propose the 3D encoder to output another

output which is transformed into a mapping layer, which is applied to the latent to discard bits in a trained fashion. The research work [47] used importance maps to directly apply bit allocation in the latent representation.

Here, we are focusing on an important issue that has not been explored as much in the literature. The construction of models specialized in outputting reconstructions at the desired target rate. A similar idea has been recently explored by Rozendaal *et al.* [48], although it is not oriented to achieve a target rate. Instead, Rozendaal's work focuses on modifying loss functions to obtain models with specific reconstruction qualities. As outlined in the next sections, a range of different problems needs to be addressed when optimizing for achieving a target.

A VAE-based image encoder trained for a specif rate remains an open problem in the literature. Solving this issue will reduce the need for training several models. Note that, this is a different problem than having a model able to achieve several rates since such a model does not necessarily yield the desired rate for specific inputs. A coding tool providing the ability to achieve a specific rate can be a deterministic factor to apply learning-based codecs in practical application scenarios. Here we focus on this issue, using an unconstrained solution that will generate some rate fluctuation. However, that fluctuation can be controlled and is way less drastic than the current state-of-the-art VAE models.

3. Motivation

The most common variational formulation of Bayesian inference relies on the Kullback-Leiber divergence [49]. A variational autoencoder (VAE) is a neural architecture that adopts this Bayesian formulation.

In the context of learning-based image compression, the seminal work [14] pointed out that a Bayesian formulation that approximates the posterior inference model may be derived from a rate-distortion Lagrangian function of the form:

$$J = \lambda \cdot D + R, \tag{1}$$

where D is the distortion between the input and reconstruction, R is the rate, λ is the Lagrangian multiplier. Equation 1 is equivalent to the rate-distortion RD function optimized in classical image encoders.

In this scenario, the optimization algorithm attempts to find the parameters for the model that minimizes a weighted sum of distortion and rate during training. Evaluating the distortion comes down to computing the chosen distortion measure between the raw input image and its reconstruction, whereas the rate is estimated with respect to the learned entropy model of the latent representation. Usually, different RD operation points and models are obtained by training the network with different values for the λ parameter, each with its own set of internal parameters.

Gradient descent optimizers rely on the differentiability of operations in the model, so quantization poses an obstacle that needs to be handled. To enable training using backpropagation, a differentiable function is used to approximate the actual quantization that takes place at the inference stage. The entropy model for the latent representation is modeled in several ways in the literature with different degrees of sophistication and computational cost.

A straightforward approach assumes latent elements within the same map are independent and identically distributed, and then fits a function to model the distribution of latent. State-of-the-art approaches exploit both hyperprior to capture spatial dependency in the latent representation at the cost of expending bits to signal side information and autoregressive context at the cost of losing parallelism. The rate for a given latent vector is estimated by computing the self-information concerning the entropy model, whereas the entropy of the latent by taking the expectation over the training dataset.

Figure 1 shows the results for Kodak dataset [18] from 8 Ballé trained models [50]. It shows the bitrate variation

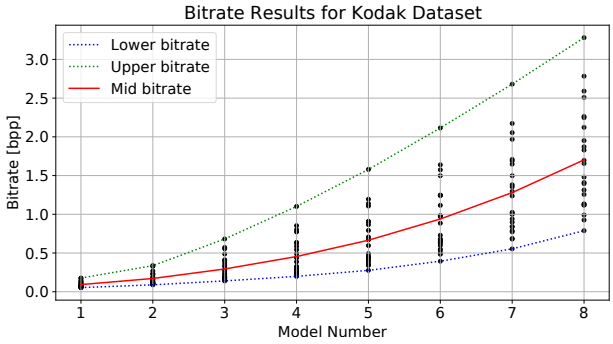


Figure 1: Results from models provided by Ballé et al. [50]. The dots represent the bitrate for each image in the Kodak dataset, for each specific model. Even though it was not trained to achieve specific bitrates, the spread of each model is quite large.

using each of the 8 pre-trained models. Since these models were not trained for specific target bitrates, they can produce a broad range of bitrates, depending on the image being encoded. Each model shows a large variance in terms of bitrate and quality. For instance, model 1 ranges from 0.053 to 0.177 bpp (3.3 times), whereas model 5 ranges from 0.276 to 1.579 (5.7 times).

If achieving a certain target bitrate is desired, a straightforward approach would be to encode each image with multiple neural models and choose the representation with the bitrate closest to the desired one. Since the models can be ordered, a binary search could be employed to avoid testing all models. However, this method has significant drawbacks. Each trained network is related to a fixed point in the RD-curve for each image and it is not possible to dynamically adjust the parameters of the baseline models to produce a different set of rates. Therefore, it is necessary to train several models and there is no guarantee that the ensemble of models would produce good bitrate matches.

Even if the desired bitrates are achieved, both the encoder and decoder would need to store a large number of trained models, each in the order of hundreds of megabytes, which would be unfeasible in many applications. Some VAE-based variable bitrate approaches [37] construct a single network that can produce many different RD-points for each image. It takes the role of the many-models approach, but it still has difficulty in achieving a specific target bitrate.

4. Proposed Method

We propose a straight yet effective modification to the usual loss function, equipping us with a tool to drive the rate operation point R resulted from training to attain a given target rate R_t. The idea is to consider the distortion loss usually applied to autoencoders and include the rate restriction using a Lagrangian relaxation approach. Lagrangian relaxation proposes the inclusion of optimization restrictions as terms in the loss. This yields a cost function that penalizes deviation from the target rate:

$$J = D + \beta \cdot f(R, R_t) \tag{2}$$

where,

$$f(R, R_t) = \left(\frac{R_t - R}{R_t} \right)^2 \tag{3}$$

We included the relaxed restriction of the target requirement as the term $f(R, R_t)$ whose relevance in the optimization is controlled by β. The greater the deviation of R from R_t, the greater the penalty to the cost function. In the extreme case where R equals R_t, the rate does not contribute to the cost function and the model attempts to minimize only the distortion.

The chosen function $f(R, R_t)$ is smooth in all its support, it also has the property of decreasing penalization of values near R_t and accounting more heavily for values far from R_t, which can be interesting for the optimization. Another characteristic of the loss is the normalization of the rate function. This normalization of the rate parameter was found to achieve a more robust function. Even though the squared version is used, any function with desired differentiability properties can be adopted.

4.1. Analysis of the proposed modification

The proposed loss function may be expressed as:

$$J = D + \beta \frac{R}{R_t}^2 - 2\beta \frac{R}{R_t} + \beta \tag{4}$$

When $R = R_t$, the loss reduces to $J = D$. Thus, whenever the target rate is satisfied only the distortion is minimized. Therefore, this loss induces local *minima* at $R \rightarrow R_t$. These local *minima* are not only influenced by $R \rightarrow R_t$ but are also determined by lower values of D in these points.

The rate term $f(\frac{R}{R_t}) = \beta \frac{R}{R_t}^2 - 2\beta \frac{R}{R_t} + \beta$ is a parabola with both roots at $R = R_t$. The β term is related to the "width" of the parabola: larger values will make it narrower and lower ones will make it wider. So, larger β will penalize more heavily the shift from the root $R = R_t$. As a consequence, it will act like a knob to control the variation of the achieved rate. The term β is also dependent on R_t as the rate of relevance will be relative to the distortion values.

This characteristic is illustrated in Figure 2 in the context of the convex-hull. For this illustration, we have adopted random points in the RD-space. The rate parabola can be thought of as a mechanism to focus desired points in the RD plane as it contributes to the value of the loss. Ideally, we would desire to reach the red point at the convex hull. However, as in any rate control mechanism, it is hard to achieve optimality.

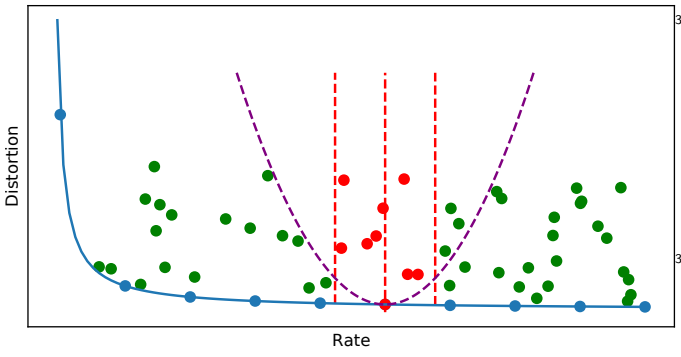


Figure 2: The parabola influences the points in RD-plane as it contributes to the distortion loss. The purple-dashed line represents the additional values that the parabola introduces in the loss function. It is establishing a tolerance interval, described by the red-dashed lines, where the central red line is at the target rate. This tolerance interval is a consequence of the fact that points outside this interval will have large loss values, and as a consequence will not likely be selected. The network can converge to any of the red points, however not necessarily to the optimal point at the convex-hull (blue line).

4.2. Training procedure to achieve the target rate

The proposed loss is defined by two hyperparameters: β and R_t. R_t is related to the mean rate of the network, and β controls the deviation from the mean rate. Both parameters need to be set taking into consideration the architecture and convergence of the network, which can be something difficult to specify in advance. β is also rate-dependent as the Lagrangian multiplier in Equation 1 for a conventional encoder. Also, there is the quantization problem in compression approaches based on neural networks, which can lead to different behavior in inference time (as it will be detailed in Section 5.3).

Hence, we devise a general training heuristic to define the hyperparameters that are being introduced (β, R_t). The objective is to avoid a large number of empiric tests that are not desired in a real application. The proposed training procedure is presented in Algorithm 1.

The inputs to the algorithm are as follow: r is the mean rate that the model should achieve (target), σ represents the allowed tolerance from r (which is modeled as a variance), t_iter is the number of iterations for training, δ_β determines the granularity of the optimization of the β parameter.

The heuristic consists of two main steps. There is an "adjustment phase" which aims to set the values of β and target R_t. In the second stage, the pre-trained network of the previous phase is further trained with the obtained $\{R_t, \beta\}$ values which should yield the desired mean rate r with the desired variance. Note that due to the mean shift issue that occurs during training R_t is not necessarily set as r (this will be detailed in Sec. 5.3).

In the adjustment phase, the objective in each step is to train the network until the rate loss is stabilized for a given $\{R_t, \beta\}$. Fig. 3 shows a plot where it is possible to see that this convergence can happen quickly after the change of the loss hyperparameters $\{R_t, \beta\}$ in a sequence of fine-tuning of a model. So, in general, a lengthy adjustment phase is not required. Nevertheless, this number

Algorithm 1: Heuristic Pseudo-code

Input : r : desired mean rate;
σ : desired variance;
t_iter : training iterations;
δ_β : granularity of change of β

Output : Trained network producing mean rate r and variance of rate σ

Initialization: $R_t = r$;
$\beta = 1$;
$R'_t \neq r$;
$\sigma' \neq \sigma$.

while $R'_t \neq r$ *and* $\sigma' \neq \sigma$ **do**
  (Re-)Train the network with $\{R_t, \beta\}$-loss in the training dataset until the rate loss becomes stable, following any desired criterion.
  Evaluate the network in the validation dataset, calculating the real compress rate obtained for each image in the dataset. Based on the set of rates obtained, compute the mean rate R'_t and variance of the rates, σ'.
  if $R'_t \neq r$ **then**
    $R_t = R_t - [R'_t - r]$
  if $\sigma' > \sigma$ **then**
    $\beta = \beta + \delta_\beta$
  if $\sigma' < \sigma$ **then**
    $\beta = max(0, \beta - \delta_\beta)$
After obtaining the $\{R_t, \beta\}$-loss, which yields a mean rate r and variance σ, train the network with this $\{R_t, \beta\}$-loss for t_iter iterations in the training dataset.

of iterations is architecture-dependent. Approaches that employ more complex architectures will naturally require a higher number of iterations. Fig. 3 represents a common training behavior of the loss during the parametric model. Each segment (represented by a different color) indicates a change in the R_t parameter. In the first segment, the output yields an average rate that is far from the desired one, this is expected since the training just started. Each R_t modification in the following segments gets the average rate closer to the desired target. The β parameter is modified within the segments as its impact in the variance the bit-rates. To detect convergence for each segment many different techniques for analyze stationarity of functions can be used [51].

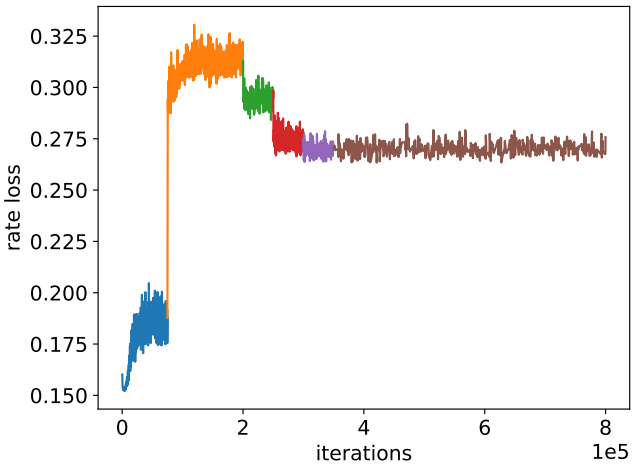


Figure 3: This figure depicts the behavior of the loss during the training of a version of the parametric model, as an example. Each color represents the fine-tuning of the model with a different set of $\{R_t, \beta\}$. The last training (in brown) is the final training and it's possible to see that the rate remains stable. It is worth highlighting that for sake of visualization, we omitted the first iteration point.

An important feature of the heuristic is that in the adjustment phase, we do not completely re-train the network for each $\{R_t, \beta\}$ pair, as already seen in Figure 3. Successive refinement steps are applied. This leads to faster convergence. Once the network is trained in this adjustment phase, it is necessary to use a validation dataset for evaluation. If the results are not satisfactory, it will be necessary to adjust R_t and β. The term R_t of the loss can be shifted and β can be increased if less variance is required or decreased otherwise. As a result of the adjustment, new parameters are obtained. Therefore, another round of fine-tuning is performed and the results are re-evaluated. This process finishes when the trained model produces a rate confined within the allowed rate fluctuation around the target rate.

The first stage of the heuristic can be seen as a procedure to define the hyperparameters of the loss. The result is a $\{\beta, R_t\}$-loss that can produce a network with the

desired mean rate r and the expected variance σ. This procedure allows applying the proposed modification to the loss function into different architectures and to obtain the new required hyperparameters ($\{\beta, R_t\}$) independently from such architectures.

One important thing to mention is the convergence of this heuristic. There is no way to mathematically guarantee that $R \to R_t$ in the heuristic with the desired variance. It relies on the neural architecture itself. As R is a variable of the loss, which also has the constant R_t as a reference, the convergence of R to R_t relies on the convergence of the network considering the loss, as in any other task for neural networks. Considering this fact, and the experimental results we will show in the result sections, we can empirically consider that the target-loss has good convergence in general and, as consequence, $R \to R_t$ in most cases. Nevertheless, if any issue in convergence occurs, a good path to follow is to try different, or more robust, architectures. Lastly, it is worth mentioning that these convergence properties can be an object of future works.

5. Baseline Architectures

To show the effectiveness of the proposed loss function, we use two different model architectures, which are essentially distinguished by the underlying entropy model. The first, which we call the *non-parametric* version was introduced in [14]. In it, the entropy model is structured using a non-parametric distribution. The second one, which we call the *parametric* version was presented in [7]. This version introduces the concept of hyperprior and models the entropy with a parametric distribution. Both architectures employ a VAE approach and are widely known, with many works proposing modifications and improvements on top of them.

5.1. Non-parametric distribution model

The non-parametric architecture used in this work was introduced in [14] and is represented in Figure 4. It is a variational autoencoder that models directly the rate-distortion trade-off in the loss. $\mathbf{x}$ is the input vector, $\tilde{\mathbf{x}}$ is the output in training time (or inference time), $\mathbb{E}$ is the expectation operator, $U|Q$ represents either the addiction of uniform noise in training time or the real quantization performed in inference time. The adopted in this figure are inspired by the work [7]. The optimization loss is defined in Equation 5, as follows:

$$L[g_a, g_s, p_{\tilde{\mathbf{y}}}] = -\mathbb{E}[\log_2 p_{\tilde{\mathbf{y}}}] + \lambda \mathbb{E}[d(\mathbf{x}, \tilde{\mathbf{x}})] \tag{5}$$

In this equation, $-\mathbb{E}[\log_2 p_{\tilde{\mathbf{y}}}]$ is the differential entropy of the latent space with distribution $p_{\tilde{\mathbf{y}}}$, and $\mathbb{E}[d(\mathbf{x}, \tilde{\mathbf{x}})]$ corresponds to the distortion. The parameter λ balances the trade-off expressed by this loss. The triplet $(g_a, g_s, p_{\tilde{\mathbf{y}}})$ is the set of elements involved in the optimization, where g_a and g_s represent the encoder and the decoder functions respectively (which are named analysis and synthesis in [14]).

The quantization taking place at inference is a non-differentiable operation. Thus, it is common to approximate the quantization operation by another operation during training to facilitate the use of stochastic gradient descent. In [14], the authors propose to use additive uniform noise as an approximation to the quantization during training. Considering $\mathbf{y} = g_a(\mathbf{x})$ as the raw latent in training time, the entropy model for the noise latent $\tilde{\mathbf{y}}$ is given by the following equation:

$$p_{\tilde{\mathbf{y}}|\psi}(\tilde{\mathbf{y}}|\psi) = \prod_i \left(p_{y_i|\psi^{(i)}}(\psi^{(i)}) * \mathcal{U}\left(-1/2, 1/2\right) \right)(\tilde{y}_i) \tag{6}$$

with vectors $\psi^{(i)}$ encapsulating the parameters of the distribution $p_{y_i|\psi^{(i)}}$ and where $\mathcal{U}(-1/2, 1/2)$ is the uniform density in the interval $[-1/2, 1/2]$. For more details about the parameters ψ, see [14] and [7], which has also an explanation of the implementation of these parameters.

It is worth to mention that $p_{\tilde{\mathbf{y}}|\psi}$ is composed of univariate distributions $p_{\tilde{y}_i|\psi^{(i)}}$ which are independent random variables. This approximation scheme, where we assume the variational family factorizes, is called mean-field ap-

proximation, and it is the most common type of variational inference as it is conceptually simple, implementation-wise easy and particularly suitable for problems involving large number of latent variables [52]. The distribution $p_{\tilde{y}_i|\psi^{(i)}}$ is a probability density function, therefore, to obtain a probability mass function $p_{\hat{\mathbf{y}}}$ suitable for use in inference time, we proceed as follows:

$$p_{\hat{y}_i}(n) = \int_{n-1/2}^{n+1/2} p_{\tilde{y}_i}(t)\,\mathrm{d}t,\ \forall n \in \mathbb{Z} \tag{7}$$

For details about constructing the mass function $p_{\hat{\mathbf{y}}}$, and the specific scheme for the use of this distribution in a real entropy coder, again consider [14] and [7]. Considering the modeling adopted in [20], the authors in [14] model the reconstruction distribution in training time, which stands for the likelihood in Bayes theorem, as:

$$p_{\mathbf{x}|\tilde{\mathbf{y}}}(\mathbf{x}|\tilde{\mathbf{y}}) = \mathcal{N}(\tilde{\mathbf{x}}, (2\lambda)^{-1}\mathbf{1})(\mathbf{x}) \tag{8}$$

where $\mathcal{N}$ stands for the Gaussian density with mean defined by the decoder output $\tilde{\mathbf{x}} = g_s(\tilde{\mathbf{y}})$ and variance determined by the inverse of λ, the Lagrangian multiplier of the loss.

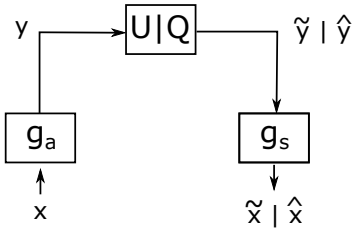


Figure 4: Diagram showing the structure of the model [14], where the arrows indicate the flow of data. g_a and g_s stand for analysis transform and synthesis transform.

5.2. Parametric distribution model with hyperprior

This architecture, depicted in Figure 5, is an extension of the work [14] and was proposed in [7]. Following many of the modeling steps described in Section 5.1, the work includes a variational hyperprior, which is also a variational autoencoder. The hyperprior is modeled using the non-parametric distribution and feed the main entropy model. In this context, the input to the analysis of the hyperprior is the latent of the main autoencoder, $\mathbf{z} = h_a(\mathbf{y})$.

A non-parametric factorized density models the variable $\mathbf{z}$ following the same approach as the Equations 6 and 7 modeled for the variable $\mathbf{y}$ in the earlier work [14]. This time, however, the main latent, also represented by variable $\mathbf{y}$, has a parametric factorized distribution, defined by:

$$p_{\tilde{\mathbf{y}}|\tilde{\mathbf{z}}}(\tilde{\mathbf{y}}|\tilde{\mathbf{z}}) = \prod_i \left(\mathcal{N}(0, \tilde{\sigma}_i^2) * \mathcal{U}\left(-1/2, 1/2\right)\right)(\tilde{y}_i) \tag{9}$$

where the variance of the Gaussian is the output of the synthesis of the hyperprior in training time, $\tilde{\sigma} = h_s(\tilde{\mathbf{z}})$. The reason why the new auxiliary VAE is called a hyperprior is due to the Bayesian inference interpretation, since $p_{\tilde{\mathbf{z}}}$ is a prior over the main prior $p_{\tilde{\mathbf{y}}}$. In other words, $p_{\tilde{\mathbf{y}}}$ is conditioned on $p_{\tilde{\mathbf{z}}}$, as explicitly shown by Equation 9 with the notation $\tilde{\mathbf{y}}|\tilde{\mathbf{z}}$ in the definition of the density $p_{\tilde{\mathbf{y}}|\tilde{\mathbf{z}}}$.

Given the introduction of the hyperprior VAE and the parametric modeling of $\mathbf{y}$ defined by a Gaussian with zero-mean and variance produced by the hyperprior, the optimization loss is different from [14], since it must also optimize the differential entropy of the hyperprior:

$$L = -\mathbb{E}[\log_2 p_{\tilde{\mathbf{y}}}] - \mathbb{E}[\log_2 p_{\tilde{\mathbf{z}}}] + \lambda\mathbb{E}[d(\mathbf{x}, \tilde{\mathbf{x}})] \tag{10}$$

The new term $-\mathbb{E}[\log_2 p_{\tilde{\mathbf{z}}}]$ is the entropy of the hyperprior. To obtain the probability mass functions $p_{\hat{\mathbf{y}}}$ and $p_{\hat{\mathbf{z}}}$, to use at inference time, the modeling of the Equation 7 is again adopted for $p_{\hat{\mathbf{z}}}$. The difference relies on $p_{\hat{\mathbf{y}}}$, as it is modeled as a factorized parametric function, the Gaussian, and is given the following integral:

$$p_{\hat{y}_i}(\hat{y}_i|\hat{\sigma}_i) = \int_{\hat{y}_i-1/2}^{\hat{y}_i+1/2} \mathcal{N}(y|0, \hat{\sigma}_i)\,\mathrm{d}y \tag{11}$$

There are two main differences between the Equation 7 and Equation 11. The first is that the integral in Equation 11 can be evaluated in closed form, as $\mathcal{N}$ has an analytic definition. Lastly, $p_{\hat{y}_i}$ must be obtained "on the fly" in this case, since it depends on $\hat{\sigma} = h_s(\hat{z})$, the output parameter of the hyperprior in inference time, which is different for each image.

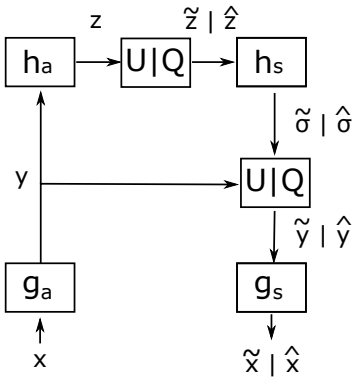


Figure 5: Structure of the parametric [7]. The arrows indicate the flow of data, the new elements are h_a and h_g, which represent the analysis and synthesis of the hyperprior, respectively.

In [8] an extension over [7] is presented with the inclusion of an auto-regressive component which combined with the hyperprior produces a Gaussian with both mean and variance as learned parameters. This Gaussian is the model of the main latent represented by variable $\mathbf{y}$. However, the authors did not present a complete implementation of this architecture due to the serial computational burden of the auto-regressive component. Instead, in the same work, the authors presented an idea of a parallel architecture where the hyperprior alone produces both parameters of the Gaussian, without an auto-regressive component. This is the parametric architecture used here to showcase the proposed solution, where the entropy model for the latent $\tilde{\mathbf{y}}$ is given by:

$$p_{\tilde{\mathbf{y}}|\tilde{\mathbf{z}}}(\tilde{\mathbf{y}}|\tilde{\mathbf{z}}) = \prod_i \left(\mathcal{N}(\tilde{\mu}_i, \tilde{\sigma}_i^2) * \mathcal{U}\left(-1/2, 1/2\right)\right)(y_i) \tag{12}$$

with $[\tilde{\mu}, \tilde{\sigma}] = h_s(\tilde{\mathbf{z}})$, the output of the hyper-synthesis in training time.

For a complete explanation of all models mentioned here and all the details of the implementation of these functions, please refer to [14, 7, 8]. Lastly, all notations adopted here follow the ones presented in [7].

5.3. Remarks on rate shift between training and inference

A heuristic was provided in Section 4 as guidance to set up values for the loss function hyperparameters to drive the rate to a specified target rate. The impact of the new hyperparameters may differ from architecture to architecture. In both baseline architectures selected to showcase the usage of the proposed solution, it was observed a mismatch between the target bitrate defined by R_t during training and the actual rate r obtained during inference. This behavior is rooted in the different ways quantization is handled during training and inference stages, it is particularly noticeable at the lower bitrates as argued below.

In the selected baseline architectures, the quantization step needs to be replaced by a differentiable approximation to allow training using backpropagation. Namely, in the baseline architecture with hyperprior, at the inference stage, each element y_i of the raw latent representation $\mathbf{y}$ is scalar quantized after subtracting the mean μ_{y_i}. In turn, the final quantized value $\hat{y}_i$ is obtained after adding the mean, which is

$$\hat{y}_i = \text{round}\left(y_i - \mu_{y_i}\right) + \mu_y \tag{13}$$

where round$(\cdot)$ rounds to the nearest integer.

On the other hand, at training, quantization is modeled as additive uniform random noise as follows

$$\text{round}\left(y_i - \mu_{y_i}\right) \approx \left(y_i - \mu_{y_i}\right) + \mathcal{U}\left(-1/2, 1/2\right) \tag{14}$$

Combining Equations 13 and 14 gives rise to the noisy latent element $\tilde{y}_i$

$$\tilde{y}_i = y_i + \mathcal{U}\left(-1/2, 1/2\right) \tag{15}$$

As the training target rate moves towards lower bitrates, the latent representations become less uncertain (low entropy). The Gaussian distributions modeling the latent elements get more and more peaked around their mean, with a narrower scale of variation. In that scenario, the difference between the estimated rate computed using the noisy latent $\tilde{\mathbf{y}}$ and the estimated rate using the quantized one $\hat{\mathbf{y}}$ becomes noticeable.

Let us consider, without loss of generality, a latent element y_i with scale parameter $\sigma_i < 1/2$ and mean $\mu_i = 0$.

The noisy latent element $\tilde{y}_i$ will (randomly) lie in $(-1, 1)$, its likelihood will be evaluated as [7]

$$\begin{aligned} p(\tilde{y}_i|\tilde{\sigma}_i) &= (\mathcal{N}(0, \tilde{\sigma}_i) * \mathcal{U}(-1/2, 1/2))\,(\tilde{y}_i) \\ &= \int_{\tilde{y}_i - 1/2}^{\tilde{y}_i + 1/2} \mathcal{N}(y|0, \tilde{\sigma}_i)\,\mathrm{d}y \\ &= \mathrm{cdf}(\tilde{y}_i + 1/2) - \mathrm{cdf}(\tilde{y}_i - 1/2) \end{aligned} \tag{16}$$

where $\mathrm{cdf}(\cdot)$ stands for the cumulative density function.

Figure 6 (left) shows schematically the computed likelihood of $\tilde{y}_i$. For $\tilde{y}_i \neq 0$, the likelihood will deviate from 1 as the integral will be evaluated in a low-density interval, resulting in the rate being overestimated during training. On the other hand, at inference, the quantized latent element $\hat{y}_i$ will be zero (due to rounding), and the likelihood will be close to 1 as schematically shown in Figure 6 (right).

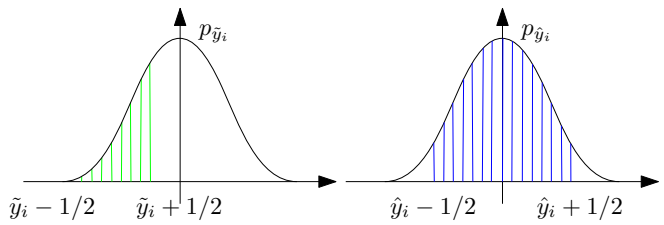


Figure 6: Difference in likelihood estimation for $\tilde{\mathbf{y}}$ and $\hat{\mathbf{y}}$

The same line of argumentation may be applied to latent elements with nonzero mean and even with other distributions. This results in rate overestimation during training, especially at low bitrates, as the useful (nonzero entropy) portion of latent is shrunk.

Approximations to overcome the issue of nondifferentiability of the quantization are applied in almost all approaches of neural coding. Different approaches to simulate this quantization in training time may lead to different behavior. It is one of the reasons why the heuristic is welcome as it can make the target-loss method robust to these mismatches produced by these approximations used in training time.

6. Experimental Results

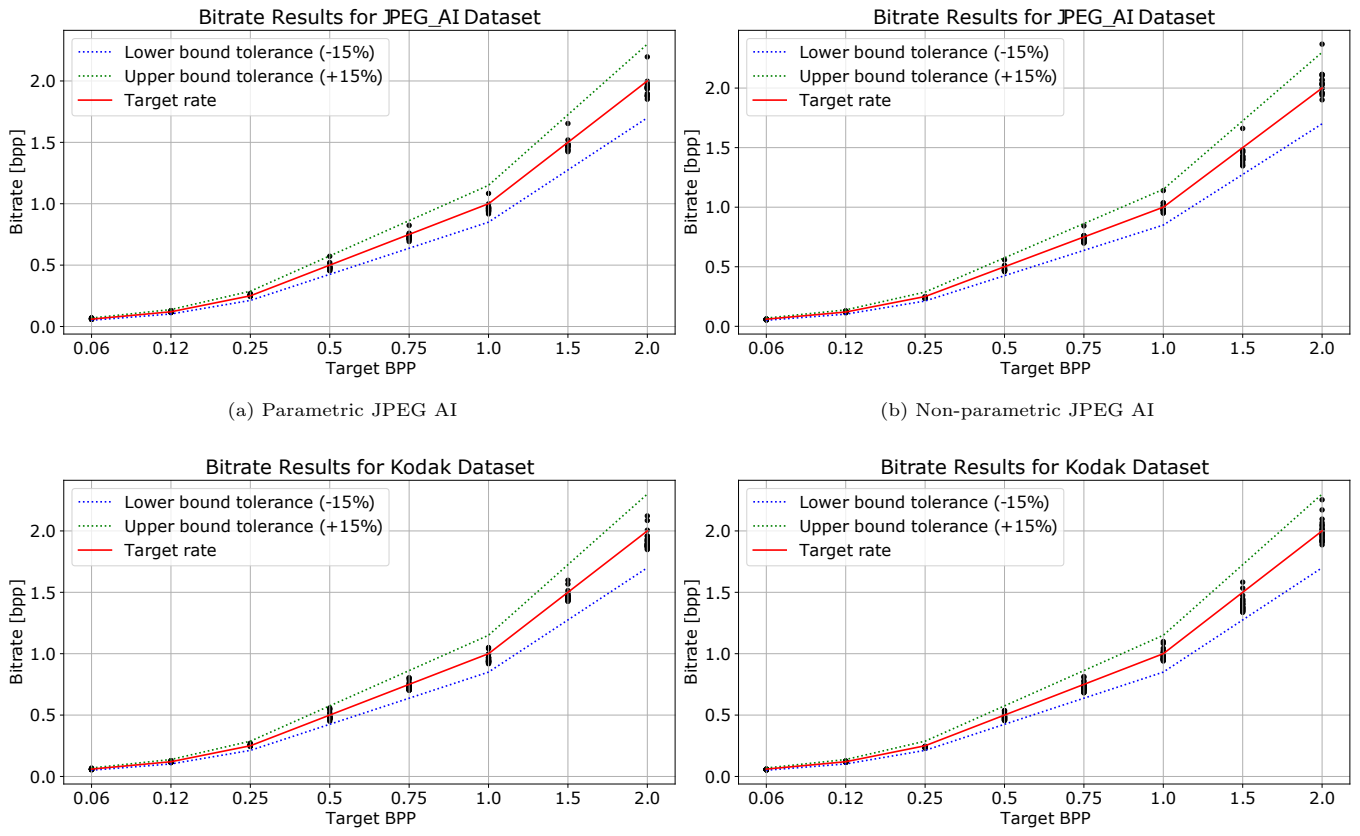
For training our models we used 256×256 non overlapping patches from the following databases: (i) CLIC Professional dataset [53]; (ii) CLIC Mobile dataset [53]; (iii) DIV2K dataset [54]; (iv) Ultra-Eye Ultra HD dataset [55]; (v) MCL-JCI [56, 57]; and (vi) FLICKR2K dataset [58].

We trained 8 networks, to reach the rates $\{0.06, 0.12, 0.25, 0.5, 0.75, 1.0, 1.5, 2.0\}$, aiming at a maximum of 15% deviation. This 15% tolerance is a challenging scenario, especially for low bitrates. Note, that the more restrictive the target bitrate, the more it may impact the RD-performance.

The *t_iter* in Sec. 4 was set to 500.000. This is the number of iterations after the pair of hyper-parameters $\{R_t, \beta\}$ is set. Since the model is completely retrained after that, the fair comparison is to set the baseline architectures to the same 500.000 iterations

The parameter δ_β (see Sec. 4), was inversely proportional to the target rate and set to an interval between $[500, 20.000]$. For lower target rates a higher δ_β results in faster convergence of the network, while for high rates with low δ_β a fast convergence can be achieved. Note, that these values were set to achieve convergence as fast as possible to reduce the time in the heuristic process for the determination of the $\{\beta, R_t\}$ pair. A different range of δ_β will only affect the convergence time frame.

The results presented in this Section were evaluated both in the JPEG AI Dataset [19] and in the Kodak dataset [18]. The Kodak dataset has been widely used as a validation dataset in many works in the literature. The JPEG AI Dataset has been recently released and the interesting point in this set of images is that they are all images with high definition, a characteristic not present in the Kodak images.



(a) Parametric JPEG AI

(b) Non-parametric JPEG AI

(c) Parametric Kodak

(d) Non-parametric Kodak

Figure 7: Target versus actual bitrate using parametric and non-parametric target loss models on Kodak and the JPEG AI datasets. These models achieve rates within the desired range (within the dotted lines) with negligible outliers.

6.1. Target rate and deviation analysis

First, it is important to show that the proposed methodology yields the expected results. Hence, using the datasets, we run similar experiments like the one provided in Figure 1. We introduced the proposed modified loss function both in the parametric and non-parametric architectures. The results are presented in Figure 7a and Figure 7b for the parametric version and Figure 7c and Figure 7d for the non-parametric architecture.

It can be observed that the vast majority of points are within the desired range (dotted lines), differently from the results shown in Fig. 1. These results suggest that the proposed loss and heuristic works well when using data that were not used during training. This means and the network can generalize to reach a specific target bitrate. An exception can be observed for the non-parametric model using the JPEG-AI dataset where one point is outside the desired range at high rates. Also, While not easy to see, there is one more point outside the range for the JPEG-AI dataset for the parametric model at the target rate of 0.06.

Considering that the JPEG AI dataset has 16 images and we are evaluating these images in 8 rates, we performed, for each architecture (non-parametric and parametric), a total of 128 reconstructions. In both cases, just a single image (1 out of 128) was outside the desired rate interval. These reconstructions have deviations of up to 3% above the upper tolerance limit. In the case of the Kodak dataset, which has 24 images, each model performed 192 reconstructions. For the Kodak dataset, no image was outside the tolerance interval. This result reinforces the generalization of the loss and the heuristic for different datasets and different architectures. Since we trained the

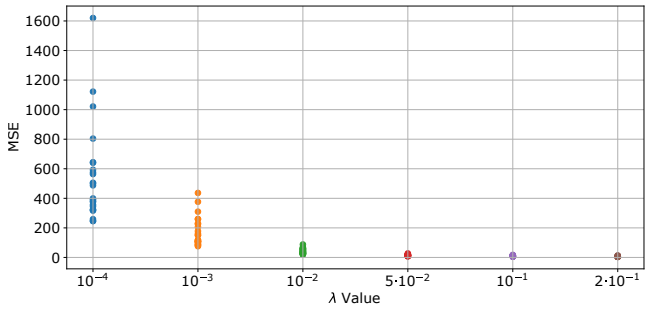


Figure 8: MSE values of the Kodak dataset for each λ model of the parametric baseline.

networks only for $500k$ iterations, one way to decrease the likelihood of getting images outside the tolerance interval would be to perform lengthier training routines (with bigger datasets as well). Another path would be to use higher β's. This would penalize more heavily deviations from the target rate, but at the cost of a higher impact on the RD performance.

6.1.1. Resnet-based experiment

One of the goals of this proposal is that the modified loss can be applied to different architectures. The baseline models selected were the seminal works in the area, named Balle's parametric and non-parametric approaches [14, 7]. Nevertheless, to test the extensiveness of the approach not only through architectures that explore different entropy models, but also through different neural operations, we devised a simple experiment on a non-parametric entropy approach which, instead of pairs of convolution + GDN, has successive residual layers, widely used in the field of deep learning, and initially proposed in [59] to solve the problem of gradients vanishing/exploding in deep architectures.

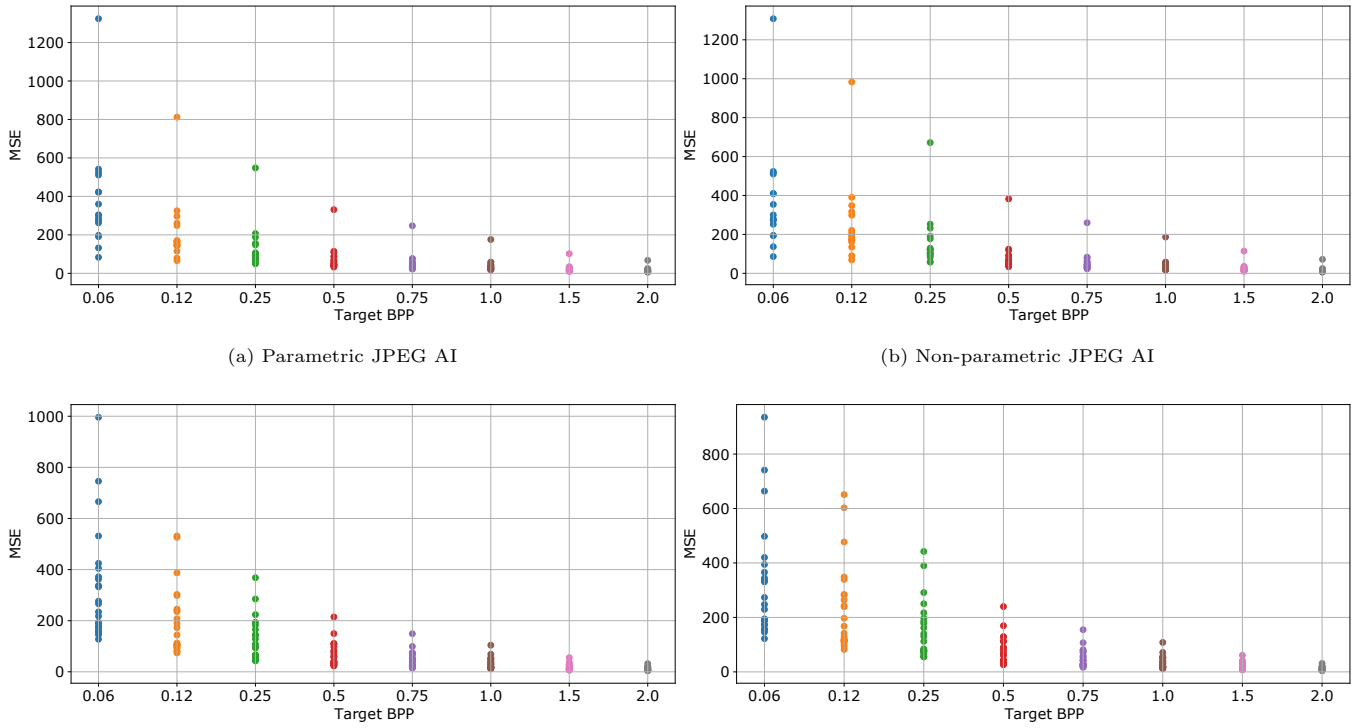
Therefore, instead of using the layers of convolutions followed by GDN's, we implemented an alternative transform that uses the same number of layers but adopting the residual blocks instead. A residual layer can be seen in Figure 10. We made the layers linear, removing the last ReLU activations, on the last layers of both the analysis and the synthesis transform, the same way it is done with the GDN convolutional layers. We have adopted 128 filters in this experiment.

Table 1: The parameters $\{R_t, \beta\}$ adopted in the residual layers transform for the non-parametric models evaluated.

r	R_t	β
0.06	0.065	55000
0.75	0.77	6000
2.0	2.0	1500

As we devised this experiment only to see if a different transform would converge to restrict the rate of the entropy model, we did not implement the full heuristic here. We set β values the same as the GDN non-parametric transform, using the values of the Table 7. We only performed some adjustments in the R_t constant to make the mean rate converge to the desired rate. Also, since it is an experiment with the focus on testing the extensiveness of the approach, we trained it only for some target rates $r = \{0.06, 0.75, 2.0\}$, which are the extreme rates explored in this work, besides an intermediate rate. The parameters for the loss are shown in Table 1. It is worth mentioning that a study of the rate deviation in the main architectures experiments will be shown later in Section 6.3, with the data shown in Table 7.

The rate results for this experiment are shown in Table 2 considering both validation datasets in the three specified rates. The mean, minimum, maximum, and standard deviation statistics of the rate are presented. These results show that even using a completely different network architecture we are still able to achieve the desired bitrate. The standard deviation in each case was very small, suggesting that lower betas could be used which, in turn, would probably improve the mean PSNR. But tuning ResNet-like architectures was not the aim of this experiment. The PSRN was only presented to show the monotonic behavior of the quality metric while increasing the rate, which is



(a) Parametric JPEG AI

(b) Non-parametric JPEG AI

(c) Parametric Kodak

(d) Non-parametric Kodak

Figure 9: Distribution of MSE scores per target bitrate for the parametric and non-parametric models. The point scattering of each bitrate demonstrates that variance decreases as the rate increases.

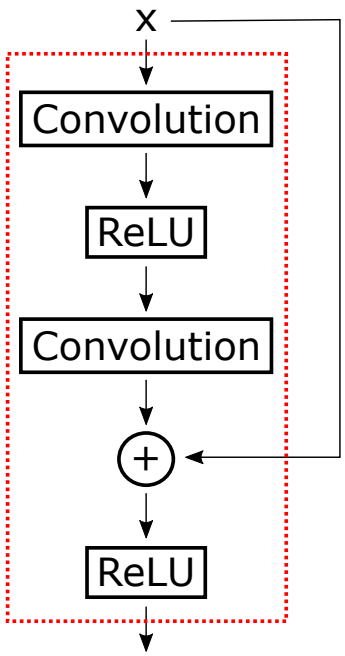


Figure 10: The building block of the residual layer. The dashed lines represents a single residual layer.

the expected behavior. Therefore, experiments with these different types of transforms will not be considered for the RD performance comparisons and may be an object of future works.

6.2. VAE-loss feature

Considering the modeling adopted in [14], where the quantization is formulated as uniform additive noise of width 1, the VAE-loss becomes the one shown in Equation 5, something highlighted in the Sub-section 5.1. As discussed in [14], not all distortions or perceptual transforms would lead to a normalizable density and, in this case, the equivalence to variational autoencoders cannot be guaranteed. The authors show, however, that any affine and invertible perceptual transform and any translation-invariant metric can correspond to a normalizable density.

Note that the proposed modification does not represent a VAE-loss. When we expand the loss as depicted in Equation 2 and Equation 3, we derive the following ex-

Table 2: The results of the residual layers experiment considering the non-parametric model for the specified target rate. It is possible to see that with the parameters specified by Table 1 it is possible to reach a good convergence on the rate on both datasets. The PSRN quality measure is shown to show that there was a monotonic behavior of the quality when increasing the rate, which is the expected behavior.

Target Rate	Kodak					JPEG AI				
	Rate				Mean PSNR	Rate				Mean PSNR
	Mean	Min	Max	Standard deviation		Mean	Min	Max	Standard deviation	
0.06	0.059	0.057	0.061	0.0008	20.77	0.059	0.057	0.067	0.0012	19.32
0.75	0.742	0.721	0.798	0.0206	31.77	0.749	0.719	0.822	0.0231	30.89
2.0	2.004	1.965	2.123	0.0423	34.95	2.030	1.958	2.208	0.0563	33.26

Table 3: The parameters $\{R_t, \beta\}$ adopted for the loss function, after the adjustment phase of the heuristic. It's noticeable that the mean shift occurred more severely in the parametric architecture. In contrast, the non-parametric model only had a discrete mean shift at low rates.

(a) Parametric model.

r	R_t	β
0.06	0.254	60000
0.12	0.275	55000
0.25	0.410	30000
0.5	0.65	9000
0.75	0.86	6500
1.0	1.08	5500
1.5	1.55	5000
2.0	1.93	2500

(b) Non-parametric model.

r	R_t	β
0.06	0.07	55000
0.12	0.145	55000
0.25	0.25	25000
0.5	0.53	9000
0.75	0.8	6000
1.0	1.08	5500
1.5	1.5	3000
2.0	2.0	1500

pression:

$$J = D + \beta \frac{R_t^2 - 2R_t R + R^2}{R_t^2} \tag{17}$$

$$J = D + \beta - \frac{2\beta}{R_t} R + \frac{\beta}{R_t^2} R^2 \tag{18}$$

This equation, as described in Section 4, correspond to a parabola, depicted in Figure 2. As β and R_t are constants, we can represent it as:

$$J = D + c_1 R + c_2 R^2 + c_3 \tag{19}$$

where c_1, c_2 and c_3 are constants. We have a term that resembles the VAE loss structure, $J = D + c_1 R$, however, in this case, it has a negative constant. There is also the additional R^2 term, which balances the negative R (considering the parabola, these terms combined with the β constant never yield a negative value). This squared term makes it hard to match a normalizable density which would lead to the proposed loss. Therefore, the correspondence to variational autoencoders cannot be guaranteed. Nevertheless, we have observed that our proposed loss hyper-parameters $\{R_t, \beta\}$ have the effect of the λ hyperparameters in Balle's loss.

The λ hyper-parameter in the VAE loss is related to the reconstruction: higher values will shrink the reconstruction variance. Lower values will allow a higher variance in the reconstruction values. This λ behavior can be seen in Figure 8, which shows the MSE values of the Kodak reconstructions for each λ-model trained for the parametric baseline. In this case, higher values of λ yield less variance in the reconstruction values.

Our proposed loss preserves some of the VAE-loss characteristics. Figure 9 shows that R_t has this same role of decreasing (or increasing) the reconstruction variance. It is the same behavior obtained when one varies the λ term in the baseline architectures.

6.3. RD Performance

In this subsection, we show results about the RD performance of the proposed methodology. Table 4 shows the

Table 4: Results for the parametric and non-parametric target models in both JPEG AI and Kodak datasets. These results are the average of "# of images" column, which indicates the number of images whose rates satisfied the interval criterion established.

(a) Parametric target in JPEG AI.

BPP	# of Images	SSIM	MSSSIM	PSNR
0.06	15	0.617	0.835	23.18
0.12	16	0.735	0.915	25.62
0.25	16	0.817	0.951	27.80
0.5	16	0.882	0.973	30.04
0.75	16	0.916	0.982	31.67
1.0	16	0.935	0.986	32.79
1.5	16	0.964	0.992	35.04
2.0	16	0.978	0.995	36.75

(b) Parametric target in Kodak.

BPP	# of Images	SSIM	MSSSIM	PSNR
0.06	24	0.609	0.832	23.73
0.12	24	0.710	0.909	26.08
0.25	24	0.797	0.949	28.30
0.5	24	0.870	0.972	30.76
0.75	24	0.909	0.982	32.53
1.0	24	0.930	0.987	33.75
1.5	24	0.961	0.993	36.42
2.0	24	0.975	0.995	38.36

(c) Non-parametric target in JPEG AI.

BPP	# of Images	SSIM	MSSSIM	PSNR
0.06	16	0.629	0.836	23.20
0.12	16	0.701	0.900	24.90
0.25	16	0.790	0.942	26.93
0.5	16	0.875	0.970	29.74
0.75	16	0.912	0.980	31.42
1.0	16	0.936	0.986	32.84
1.5	16	0.959	0.991	34.77
2.0	15	0.978	0.995	37.44

(d) Non-parametric target in Kodak.

BPP	# of Images	SSIM	MSSSIM	PSNR
0.06	24	0.611	0.833	23.82
0.12	24	0.678	0.896	25.40
0.25	24	0.768	0.939	27.32
0.5	24	0.863	0.969	30.30
0.75	24	0.906	0.981	32.17
1.0	24	0.933	0.987	33.74
1.5	24	0.956	0.992	35.86
2.0	24	0.974	0.995	38.33

results in the Kodak dataset and JPEG AI Dataset for the parametric and non-parametric target models. The results in each line are the average obtained in the dataset considering the number of images in the column "# of Images". Note, that only the images within the desired rate are considered. For the Kodak dataset, all 24 images for all target rates are always within the range, while as mentioned in Sec. 6.1 there are two instances where one of the 16 images is outside the range.

An interesting observation in these results is that the performance of the parametric model was almost equivalent to the non-parametric model. However, when the parametric model was originally proposed in [7], it showed better results than the non-parametric version originally presented in [14]. This means that there was more deterioration in the parametric target with respect to the parametric baseline than in the non-parametric pair. Nevertheless, we empirically found that the mean-shift problem formally analyzed in subsection 5.3, which is related to the quantization approximation, affected more the parametric model than the non-parametric model.

The impact of mean-shift can be seen in Table 7, which shows the $\{R_t, \beta\}$ parameters obtained after the adjustment phase described in 4.2. It can be seen that at low rates for the parametric model the mean-shift problem is very significant. As an example, in order to obtain a real

Table 5: Comparison of parametric models in both datasets, JPEG AI and Kodak. The number of common images, whose rate matched the restrictions of the target rate for both the target and baseline models, is given by the column "# of Images".

(a) Parametric models comparison in JPEG AI.

BPP	**# of Images**	**SSIM**		**MSSSIM**		**PSNR**	
		Baseline	**Target BPP**	**Baseline**	**Target BPP**	**Baseline**	**Target BPP**
0.06	4	0.736	0.660	0.888	0.849	27.07	24.16
0.12	2	0.737	0.701	0.920	0.912	24.22	23.21
0.25	5	0.936	0.898	0.981	0.972	33.28	29.60
0.5	3	0.934	0.908	0.987	0.982	32.71	29.46
0.75	6	0.975	0.960	0.995	0.990	37.96	33.50
1.0	12	0.948	0.929	0.989	0.984	36.45	33.03
1.5	12	0.972	0.958	0.994	0.991	37.47	34.88
2.0	6	0.979	0.982	0.995	0.995	36.25	34.93

(b) Parametric models comparison in Kodak.

BPP	**# of Images**	**SSIM**		**MSSSIM**		**PSNR**	
		Baseline	**Target BPP**	**Baseline**	**Target BPP**	**Baseline**	**Target BPP**
0.06	9	0.756	0.703	0.902	0.872	27.83	25.06
0.12	3	0.614	0.618	0.870	0.885	24.19	23.67
0.25	10	0.876	0.840	0.968	0.958	33.52	30.53
0.5	6	0.894	0.871	0.976	0.973	33.65	30.58
0.75	13	0.932	0.917	0.987	0.984	36.45	33.38
1.0	16	0.949	0.933	0.990	0.987	38.19	34.46
1.5	20	0.964	0.960	0.993	0.993	38.46	35.83
2.0	11	0.974	0.973	0.995	0.995	38.23	36.19

rate of $r = 0.06$ the hyperparameter R_t was set as 0.254. At higher rates, the mean-shift is less significant. On the other hand, the non-parametric model does not seem to be affected by mean-shift. These results support the analysis described in sec. 5.3, since the parametric model adopts a Gaussian distribution. A detailed study of this issue can be the object of future works.

The second experiment performed has the objective of comparing the amount of deterioration that the restrictions in the target models had on the rate-distortion of the images. We have trained 6 models of each baseline architecture (parametric and non-parametric), with different λ values, trying to cover from low-quality reconstructions to high-quality reconstructions. The λ were determined in order to try to achieve rates as similar as possible as the desired ones. These baseline models were also trained for $500k$ iterations for a fair comparison.

These models will yield different distortion and rate results depending on the input image. Nevertheless, to perform fair comparisons we adopted specific criteria for the comparisons. We encoded/decoded each image using all baseline models. If any of the reconstructions of the base-

Table 6: Comparison of non-parametric models in both datasets, JPEG AI and Kodak. The number of common images, whose rate matched the restrictions of the target rate for both the target and baseline models, is given by the column "# of Images".

(a) Non-parametric models comparison in JPEG-AI.

BPP	**# of Images**	**SSIM**		**MSSSIM**		**PSNR**	
		Baseline	**Target BPP**	**Baseline**	**Target BPP**	**Baseline**	**Target BPP**
0.06	3	0.679	0.664	0.866	0.861	27.65	26.94
0.12	5	0.734	0.699	0.904	0.901	24.68	23.80
0.25	1	0.705	0.697	0.917	0.915	31.156	30.54
0.5	4	0.853	0.860	0.960	0.963	30.82	29.82
0.75	6	0.934	0.925	0.985	0.982	35.07	32.53
1.0	5	0.940	0.944	0.988	0.988	34.23	32.56
1.5	15	0.964	0.960	0.992	0.991	37.38	35.25
2.0	10	0.970	0.972	0.994	0.994	37.64	36.70

(b) Non-parametric models comparison in Kodak.

BPP	**# of Images**	**SSIM**		**MSSSIM**		**PSNR**	
		Baseline	**Target BPP**	**Baseline**	**Target BPP**	**Baseline**	**Target BPP**
0.06	2	0.721	0.702	0.881	0.875	27.62	26.87
0.12	3	0.513	0.500	0.826	0.842	22.82	22.21
0.25	1	0.912	0.887	0.974	0.965	34.58	30.58
0.5	5	0.837	0.845	0.960	0.965	30.37	29.31
0.75	9	0.920	0.921	0.984	0.984	34.99	32.98
1.0	11	0.932	0.935	0.986	0.987	36.78	34.88
1.5	21	0.962	0.958	0.993	0.992	38.75	36.44
2.0	16	0.968	0.973	0.994	0.994	37.87	37.39

lines is within the tolerance interval of any target rate, we considered it for comparisons. Therefore, we are considering the 6 trained models of each baseline architecture as a single ensemble model capable of producing 6 different reconstructions for each image. If any reconstruction fits the requirements, it is considered valid for comparisons. This ensemble of 6 models is compared to just a 1 trained model for each rate of the proposed methodology. For example, suppose that in a specific dataset using all 6 baseline models only 1 images is within the acceptable range for 0.06 bpp, then only that image is valid for comparison. Note, that the image needs to achieve the target rate using the proposed models as well, which occurred in 100% of all cases.

With such a strategy, we can compare the average results. Tables 5 and 6 show the results for all comparisons considering SSIM, MS-SSIM, and PSNR metrics. The number of images considered following the comparison criteria is specified by the columns "# of Images".

In all sub-tables, it is clear that the quality loss for MS-SSIM and SSIM are very minor (except for the lowest bitrate). The gap is more significant for the PSNR metric.

Table 7: The parameters $\{R_t, \beta\}$ adopted for the loss function (after the adjustment phase). The mean shift occurs more severely in the parametric architecture. In contrast, the non-parametric model only presented a discrete mean shift at low rates.

(a) Parametric model.

r	R_t	β
0.06	0.254	60000
0.12	0.275	55000
0.25	0.410	30000
0.5	0.65	9000
0.75	0.86	6500
1.0	1.08	5500
1.5	1.55	5000
2.0	1.93	2500

(b) Non-parametric model.

r	R_t	β
0.06	0.07	55000
0.12	0.145	55000
0.25	0.25	25000
0.5	0.53	9000
0.75	0.8	6000
1.0	1.08	5500
1.5	1.5	3000
2.0	2.0	1500

However, it is known that MS-SSIM and SSIM are more correlated with perceptual quality [60, 61]. Also, there is an increasing interest in evaluations of neural architectures using perceptual metrics than the classic PSNR. For example, it can be noted that in the JPEG AI Call for Evidence [19], the PSNR was not even considered.

It is worth noting that we did not perform any specific training considering a specific quality metric, and MSE was set as the distortion measure for all trained models. Nevertheless, neural networks tend to produce images with high perceptual quality even when they are not trained specifically for such metrics, as described in Subsection 6.5. Therefore, we have chosen the mean squared error to also evaluate the PSNR comparisons while having high quality in the perceptual metrics.

As expected, even using reconstructions of any of the 6 baseline models, only a few of them are good matches for the target rates, which shows that rate constrained coding is an important issue to be addressed. On the other hand, as shown in Tables 5 and 6, our proposal can achieve good RD results for each rate using only one trained model.

The results of Tables 5 and 6 also highlight the higher relative deterioration between the parametric pair, than the non-parametric pair. Even though the perceptual metrics show almost no significant deterioration, as will be seen in the images presented in Sub-section 6.5, it is possible to compare the deterioration using the PSNR as a reference. And these results suggest that for some reason, the parametric model not only had more deterioration but also exhibited a higher mean-shift (as presented in Subsection 5.3).

6.4. Additional complexity of the proposed heuristics

One of the main aspects leveraged by this work is the bitrate shift that occurs between training and inference, which is discussed in Section 5.3. As we both formally and empirically have observed this happens due to the approximation of the quantization via additive uniform noise. This will require an intensive empirical test to set the hyper-parameters β and R_t. Therefore, we proposed the heuristics depicted in the Algorithm 1. The idea of the heuristics is to correct the mean bitrate through shifts of R_t parameter and increase or decrease the variance of rate through changes $\pm\delta_\beta$ in the parameter β.

When we consider the computational complexity, the main aspect to consider is that it is a straightforward approach to correct the parameters which rely on additional training and validation steps. So there is no significant increase in the computational complexity of a normal training and validation procedure, as it is a linear algorithm on these steps.

Therefore, the added computational load is the iteration increase burden introduced when adopting the steps to estimate the $\{R_t, \beta\}$ pair before actually training the neural network. In the pre-training stage of the heuristics, the pair $\{R_t, \beta\}$ is adapted at each 50.000 iterations. In our experiments, the average number of total iterations required was 220.000, with a minimum of 100.000 and a maximum of 500.000.

After this initial training, the target bitrate will be

achieved within a range controlled by β. Then the network will be completely retrained for 500.000 iterations with the obtained parameters. More iterations can be beneficial. Nevertheless, we empirically verify that this is the lowest number of iterations in which competitive RD results can be obtained for the baseline networks and ours.

One interesting thing to highlight is that not only the $\pm\delta_\beta$ has a huge impact on the length of training but also the initial value of β. In the raw Algorithm 1, we set the initial value of $\beta = 1$, but a different value may be adopted based on prior knowledge of the architecture. We are presenting the results assuming no initial knowledge on how fast the network converges. Also, without the initial training for selection of the loss parameters, several full training cycles of around $500k$ iterations would be required to set those parameters empirically. Hence, the heuristics avoid a more time-consuming process.

6.5. *Subjective evaluation*

In Figure 11 and 15 we show some reconstructions obtained using the parametric models for each data set, respectively. Each line represents one of the images whose reconstructions were inside the tolerance interval for both the baseline and proposed model. Similarly, Figures 13 and 17 some examples are shown for the non-parametric models. A complete subjective test can be the subject for future work. Here, we included in the figures the SSIM, MS-SSIM, and VMAF metrics. The VMAF metrics are considered to be highly correlated to the human visual system [62]. In the examples, we show the images with the biggest difference in terms of VMAF when we outperformed the baseline and the biggest difference when we underperformed. As it can be seen the difference in the metrics is normally small for the SSIM ad MS-SSIM metrics. For the VMAF metrics our worst results are in the first two rows of Fig. 15 and the second row of Fig.17. Our best results are comparing the VMAF metric with the baseline are the third line of Fig. 11 and the second line of Fig. 13.

RD curves for all evaluated metrics for each of the images obtained using the parametric models can be found in Figs. 12 and 16 for each data set, respectively. The RD curves for the non-parametric are shown in Figs. 14 and 18. As it can be seen, even though there is a loss in PSNR, for SSIM and MS-SSIM metrics, our proposed method yields a very competitive RD performance compared to the baseline architectures. It actually outperforms the baseline models at mid to low rates for some images. A significant drop in PSNR for rate constrained coding is expected as it is present in such tools even for conventional encoders, e.g. VVC [63].

6.6. *Spatial bit allocation*

A fundamental difference between the baseline architectures rests in their modeling of the latent representation, which in turn produces effects on the learned nonlinear transforms, and in the resulting capability of distributing the bit budget spatially. The non-parametric model assumes the latent representation follows a fully factorized density, whereas the parametric model accounts for the coupled variation between the latent elements giving it the ability to reduce the bit expenditure.

In order to meet the target bitrate, the proposed loss function penalizes deviation from the target rate and treats deviations below and above the target equally. This raises a question about the effect of the proposed loss function on bit allocation. Figure 19 shows the self-information averaged over each spatial location of the latent representation for both non-parametric and parametric models. It evidences that the network preserves its spatial bit allocation capability. Both models are able to vary the bit expenditure according to texture and structural complexity. Despite that, when assessing reconstruction quality using PSNR, the models trained to meet a target rate present a performance drop for images exhibiting large smooth areas, see Figures 12, 14, 16 and 18. We attribute this performance drop to the lack of a tool enabling end-to-

end autoencoder solutions to easily move over the rate-distortion trade-off curve. Even if the model saves bit rate in smooth areas to spend over textured/structured areas, the model will struggle to buy more quality. Except for a few works such as [10, 6], most end-to-end autoencoder solutions train a set of models to achieve a few rate-distortion operational points with no means to move in between them. Recent works [37, 64, 65] have attempted to fill the gap, however without a tool to meet a target bitrate. It is worth stressing that an alternative to achieve a target bitrate, would involve training several models to obtain enough granularity of rate-distortion operational points, and then search for the model that meets the target rate on an image basis.

Figure 19 shows the parametric model produces a spatial bit allocation with higher variance compared to the non-parametric model, as indicated by the standard deviation. This indicates the parametric model is taking advantage of the hyperprior, which aims at accounting for the spatial dependence within the latent representation, to better allocate the bit budget.

7. Conclusions

This paper proposes a solution for rate constrained encoding, an open problem in learning-based image codecs. We propose a $\{\beta, R_t\}$-loss which takes into account the mean target rate and the desired variance. To devise the parameters for the loss, a heuristic is proposed. This heuristic along with the $\{\beta, R_t\}$-loss can be applied to various codecs based on CNNs. Rate control is a desirable feature in practical scenarios. The proposed methodology may help to introduce an image codec based on VAE architecture in real applications. As with any image encoder, the RD performance is affected when a rate-control mechanism is introduced. Nevertheless, the proposed method presents minimal quality losses in terms of MS-SSIM and SSIM metrics. For some images, an RD gain can be obtained for low to mid rates for such objective metrics. Moreover, we presented some visual examples to show that there is no big perceptual difference to be noticed.

The design of a rate-target training also highlighted the mismatch caused by the approximation of the quantization operation in training time. It became clear with the mean-shift problem, which inspired the adoption of the heuristic to obtain the loss hyperparameters. We concluded this misalignment between the estimated rate in training time and the real bitrate in inference time is related to the additive random uniform noise, which causes an overestimation of the rate and, by consequence, an over-penalization of the rate during training time.

There is room for many improvements in the target-rate loss approach. For instance, a more detailed study of the impact caused by different loss functions on the RD deterioration can be done. It may lead to architectures that have small or even no deterioration on the baseline architectures.

The use of different quantization strategies can also be studied to mitigate (or even fix) the mean-shift problem. Obtaining efficient quantization approximations is, by itself, an open problem in the field. But a great benefit in our approach is to align the hyperparameters of the loss with the real obtained values decreasing the need for the heuristic.

As hypothesized in the paper, there may be a correlation between the deterioration of performance and the mean-shift problem. Future works studying this hypothesis in the target-rate approach or general approaches can bring many insights. If there is some correlation, investigating the quantization impact on the performance of the networks through the mean-shift may guide better approximation strategies to improve the performance of neural compression in general.

Applying the proposed modification in variable-bitrate architectures may also constitute a future work. Instead of the model representing multiple unknown points in the RD-curve, we could achieve multiple target BPP's with

Figure 11: Obtained reconstructions using the **parametric** models for images of the JPEG AI dataset. Original pristine (left), baseline (middle), and parametric target-loss (right). The rates considered for the reconstructions are $\{0.06, 0.12, 0.5, 1.0, 1.5\}$, representing each line of images, respectively.

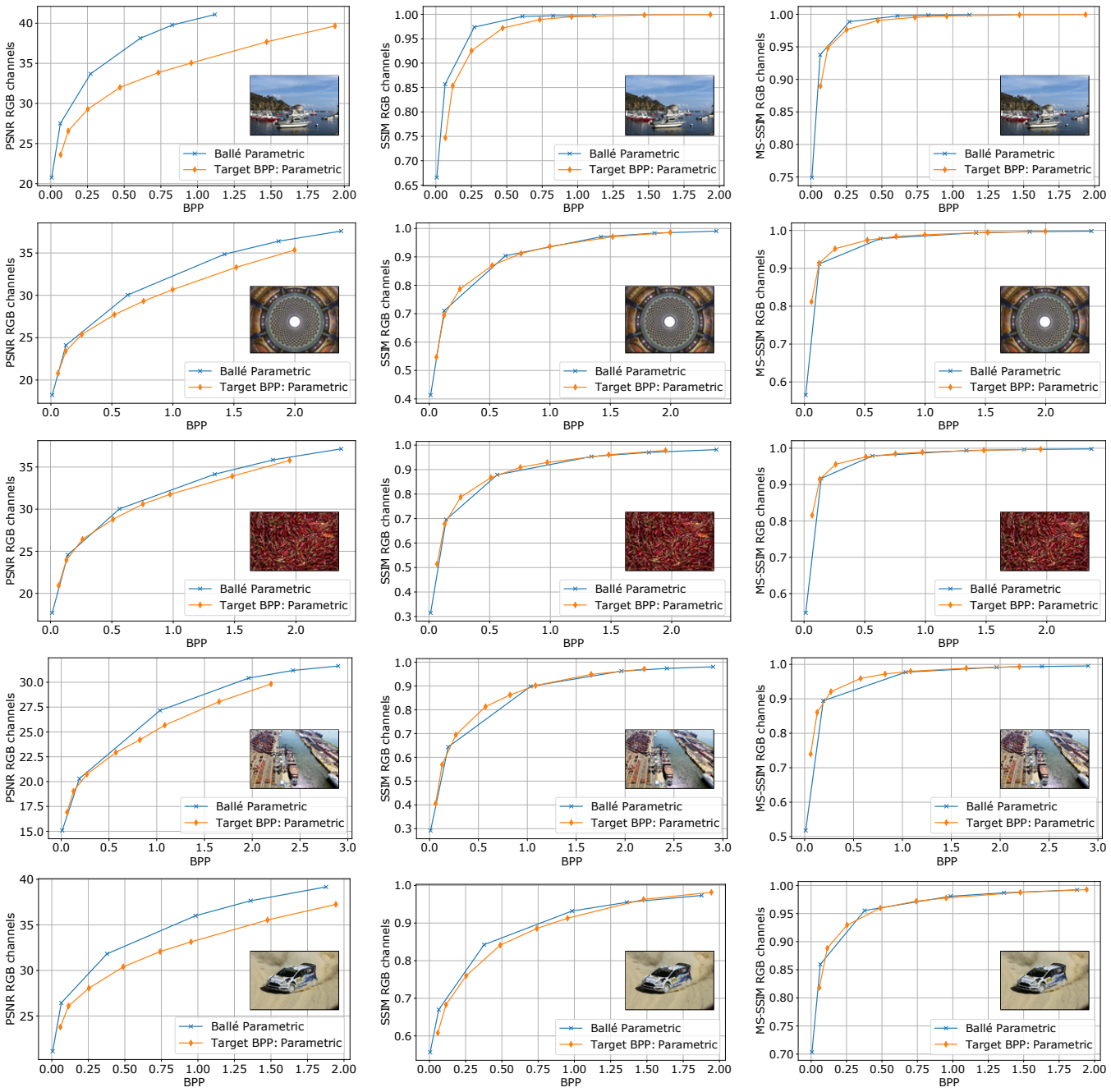


Figure 12: The RD-curves for the **parametric** models, related to Figure 11 reconstructions, using PSNR (left), SSIM (middle), and MS-SSIM (right) metrics.

this single model.

Funding

This work was supported by the *Deep Codec* project funded by Samsung Eletrônica da Amazônia Ltda under the Brazillian Informatics Law 8.248/91; BM thanks CNPq PQ 308548/2018-3.

References

[1] G. K. Wallace, The JPEG Still Picture Compression Standard, IEEE transactions on consumer electronics 38 (1) (1992) xviii–xxxiv.

[2] A. Skodras, C. Christopoulos, T. Ebrahimi, The JPEG 2000 Still Image Compression Standard, IEEE Signal processing magazine 18 (5) (2001) 36–58.

[3] F. Dufaux, G. J. Sullivan, T. Ebrahimi, The JPEG XR Image

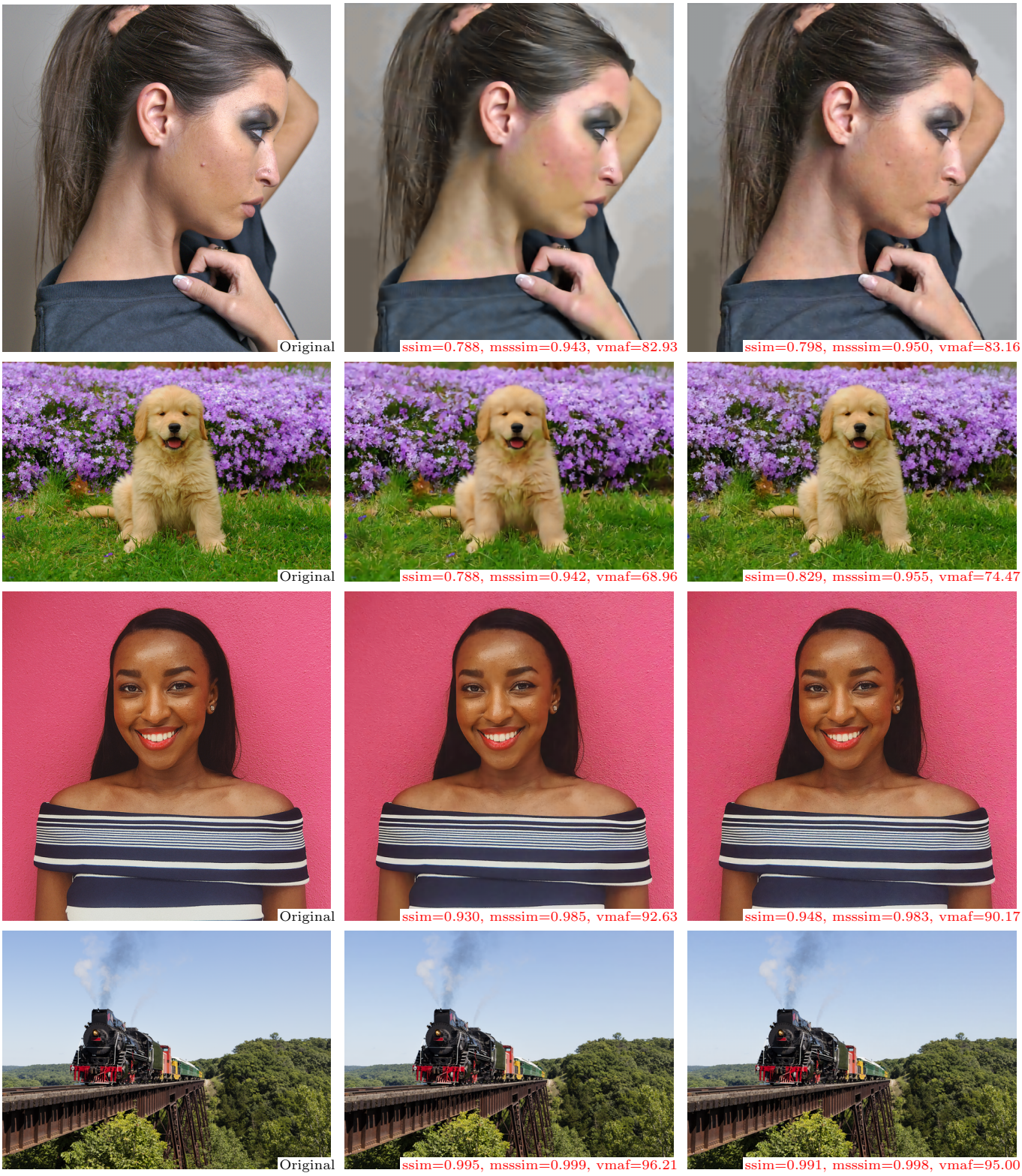


Figure 13: Obtained reconstructions using the **non-parametric** models for images of the JPEG AI dataset. Original pristine (left), baseline (middle), and non-parametric target-loss (right). The rates considered for the reconstructions are $\{0.06, 0.12, 0.5, 1.5\}$, representing each line of images, respectively.

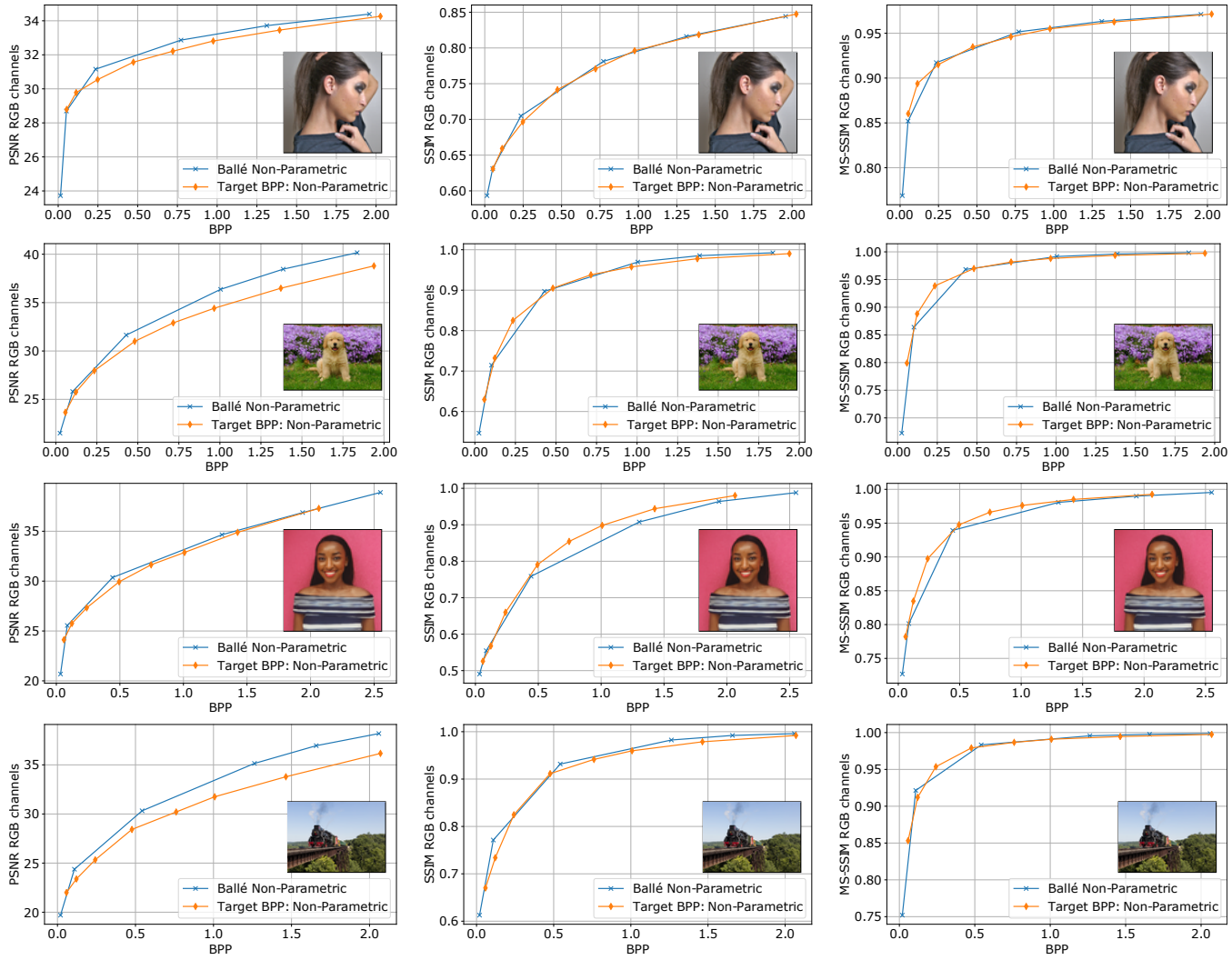


Figure 14: The RD-curves for the **non-parametric** models, related to Figure 13 reconstructions, using PSNR (left), SSIM (middle), and MS-SSIM (right) metrics.

Coding Standard [Standards in a Nutshell], IEEE Signal Processing Magazine 26 (6) (2009) 195–204.

[4] A. Artusi, R. K. Mantiuk, T. Richter, P. Hanhart, P. Korshunov, M. Agostinelli, A. Ten, T. Ebrahimi, Overview and Evaluation of the JPEG XT HDR Image Compression Standard, Journal of Real-Time Image Processing 16 (2) (2019) 413–428.

[5] J. Alakuijala, R. van Asseldonk, S. Boukortt, M. Bruse, Z. Szabadka, I.-M. Comșa, M. Firsching, T. Fischbacher, E. Kliuchnikov, S. Gomez, et al., JPEG XL Next-generation Image Compression Architecture and Coding Tools, in: Applications of Digital Image Processing XLII, Vol. 11137, International Society for Optics and Photonics, 2019, p. 111370K.

[6] G. Toderici, D. Vincent, N. Johnston, S. Jin Hwang, D. Minnen, J. Shor, M. Covell, Full Resolution Image Compression with Recurrent Neural Networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5306–5314.

[7] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, N. 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, OpenReview.net, 2018.
URL `https://openreview.net/forum?id=rkcQFMZRb`

[8] D. Minnen, J. Ballé, G. D. Toderici, Joint Autoregressive and Hierarchical Priors for Learned Image Compression, in: Advances in Neural Information Processing Systems, 2018, pp. 10771–10780.

[9] E. Agustsson, M. Tschannen, F. Mentzer, R. Timofte, L. V. Gool, Generative Adversarial Networks for Extreme Learned Image Compression, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 221–231.

Figure 15: Obtained reconstructions using the **parametric** models for images of the Kodak dataset. Original pristine (left), baseline (middle), and parametric target-loss (right). The rates considered for the reconstructions are $\{0.12, 0.25, 0.5, 0.75, 1.5\}$, representing each line of images, respectively.

[10] G. Toderici, S. M. O'Malley, S. J. Hwang, D. Vincent, D. Minnen, S. Baluja, M. Covell, R. Sukthankar, Variable Rate Image Compression with Recurrent Neural Networks, in: International Conference on Learning Representations (ICLR), 2016.

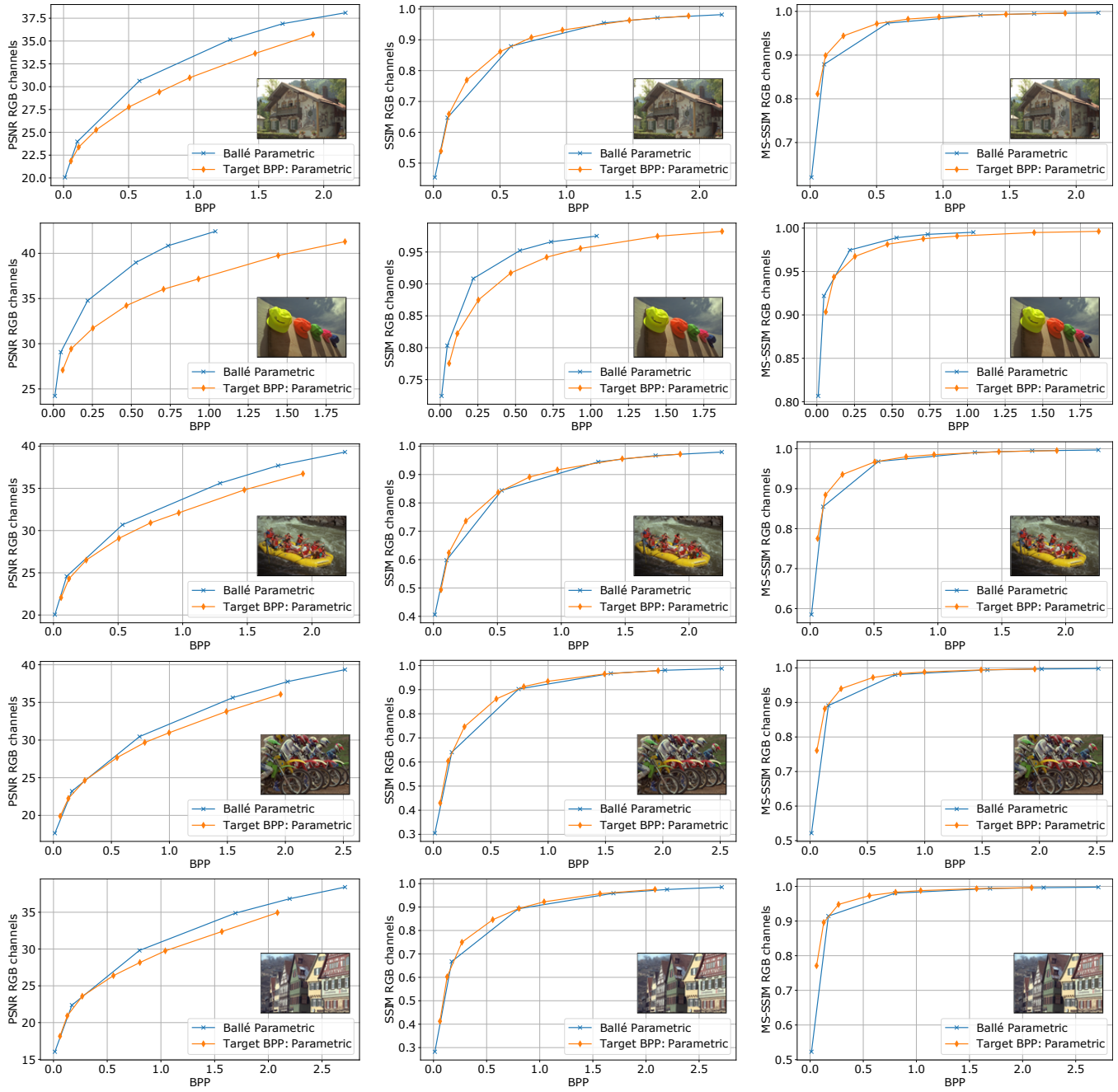


Figure 16: The RD-curves for the **parametric** models, related to Figure 15 reconstructions, using PSNR (left), SSIM (middle), and MS-SSIM (right) metrics.

[11] D. Minnen, G. Toderici, M. Covell, T. Chinen, N. Johnston, J. Shor, S. J. Hwang, D. Vincent, S. Singh, Spatially Adaptive Image Compression using a Tiled Deep Network, in: International Conference on Image Processing (ICIP), 2017. arXiv:1802.02629, doi:10.1109/ICIP.2017.8296792.
URL http://arxiv.org/abs/1802.02629

[12] H. C. Jung, N. D. Guerin Jr, R. S. Ramos, B. Macchiavello, E. Peixoto, E. M. Hung, T. Campos, R. C. Silva, V. Testoni, P. G. Freitas, Multi-mode Intra Prediction for Learning-based Image Compression, in: International Conference on Image Processing (ICIP), 2020.

[13] J. Ballé, V. Laparra, E. P. Simoncelli, Density Modeling of Images using a Generalized Normalization Transformation, in: Y. Bengio, Y. LeCun (Eds.), 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings, 2016.
URL http://arxiv.org/abs/1511.06281

[14] J. Ballé, V. Laparra, and E. P. Simoncelli, End-to-end Opti-

Figure 17: Obtained reconstructions using the **non-parametric** models for images of the Kodak dataset. Original pristine (left), baseline (middle), and non-parametric target-loss (right). The rates considered for the reconstructions are $\{0.12, 0.25, 0.5, 0.75, 1.5\}$, representing each line of images, respectively.

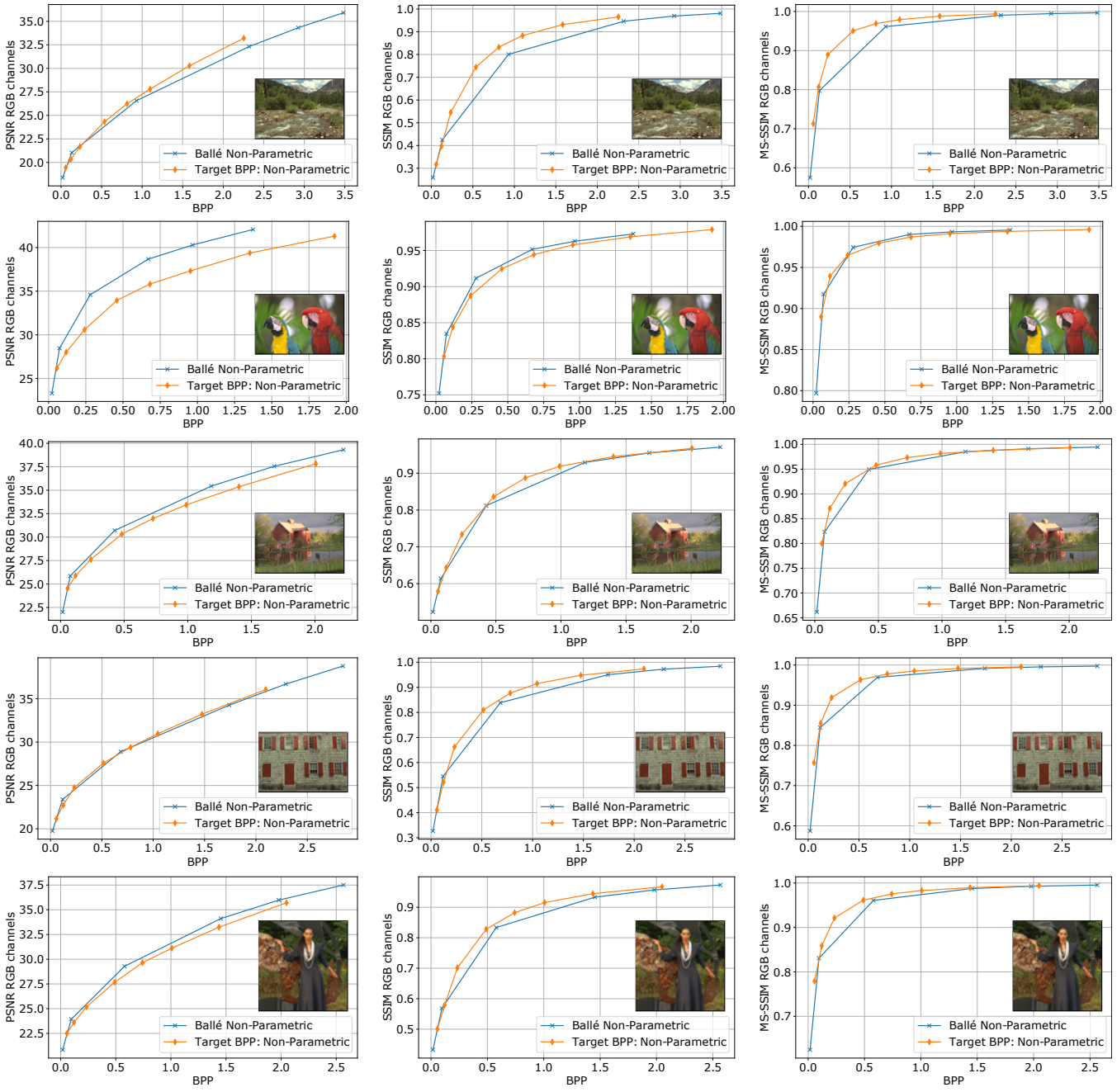


Figure 18: The RD-curves for the **non-parametric** models, related to Figure 17 reconstructions, using PSNR (left), SSIM (middle), and MS-SSIM (right) metrics.

mized Image Compression, in: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings, OpenReview.net, 2017.

URL `https://openreview.net/forum?id=rJxdQ3jeg`

[15] P. Akyazi, T. Ebrahimi, Learning-Based Image Compression using Convolutional Autoencoder and Wavelet Decomposition, in: IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2019, Long Beach, CA, USA, June 16-20, 2019, Computer Vision Foundation / IEEE, 2019, p. 0.

[16] H. Akutsu, T. Naruko, End-to-End Learned ROI Image Compression, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2019.

[17] C. Aytekin, F. Cricri, A. Hallapuro, J. Lainema, E. Aksu, M. Hannuksela, A Compression Objective and a Cycle Loss for Neural Image Compression, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops,

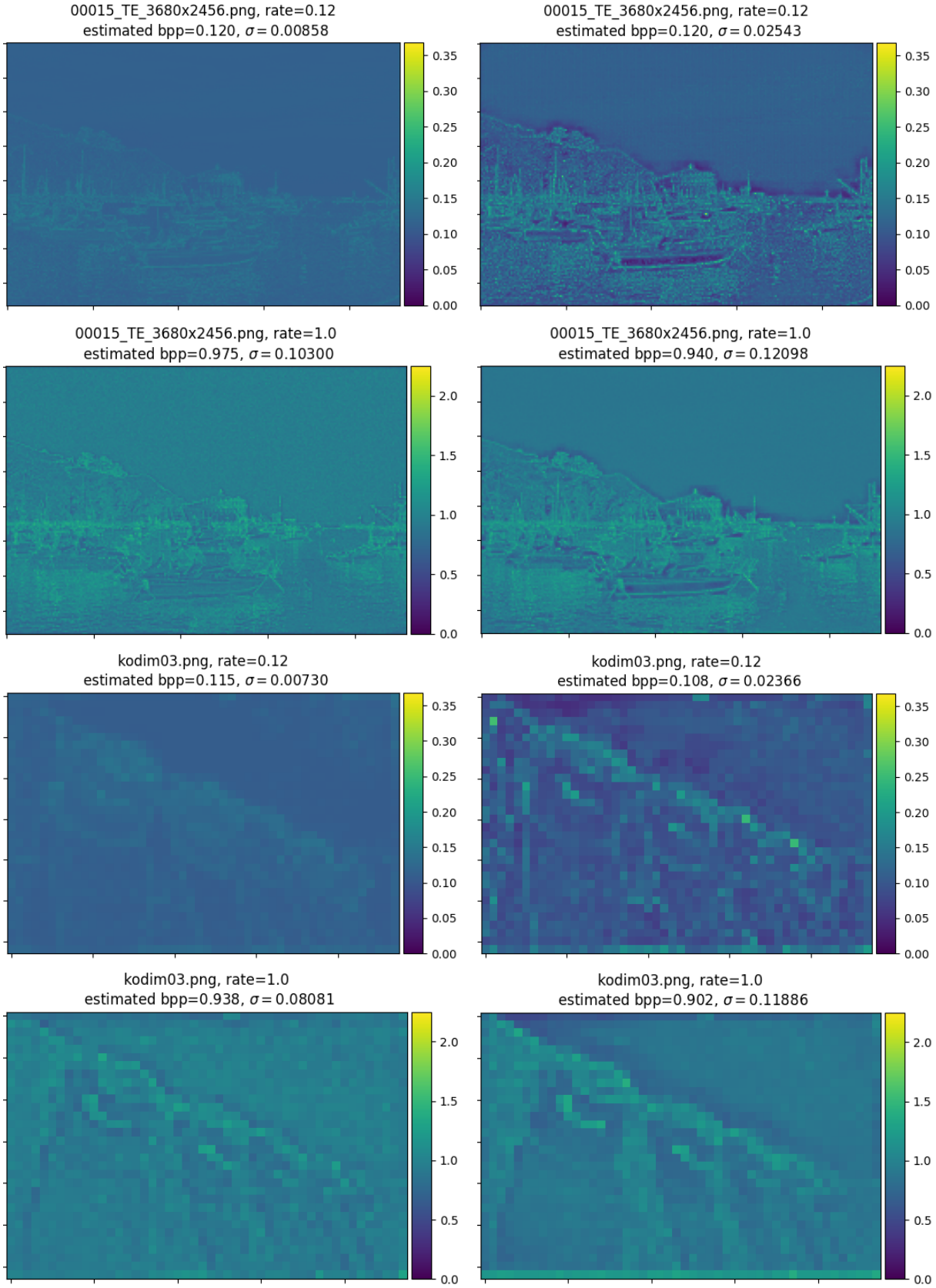


Figure 19: Spatial bit allocation maps at 0.12 bpp and 1 bpp for images *images 00015_TE_3680x2456* and *kodim03*. On the left are bit allocation maps for the **non-parametric** model and on the right for the **parametric** model.

2019.

[18] R. W. Franzen, Kodak image dataset, available from http://r0k.us/graphics/kodak/ (Updated on January 27 2013).

[19] JPEG-AI Call for Evidence - IEEE MMSP2020 Challenge (Dataset), available from https://jpegai.github.io/test_images/ (2020).

[20] D. P. Kingma, M. Welling, Auto-encoding variational bayes, in: International Conference on Learning Representations (ICLR), 2014.

[21] L. Theis, W. Shi, A. Cunningham, F. 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, OpenReview.net, 2017.
URL https://openreview.net/forum?id=rJiNwv9gg

[22] D. Minnen, G. Toderici, S. Singh, S. J. Hwang, M. Covell, Image-dependent local entropy models for learned image compression, in: 2018 25th IEEE International Conference on Image Processing (ICIP), 2018, pp. 430–434. doi:10.1109/ICIP.2018.8451502.

[23] M. Li, K. Ma, J. You, D. Zhang, W. Zuo, Efficient and effective context-based convolutional entropy modeling for image compression, IEEE Trans. Image Process. 29 (2020) 5900–5911. doi:10.1109/TIP.2020.2985225.
URL https://doi.org/10.1109/TIP.2020.2985225

[24] J. Zhou, S. Wen, A. Nakagawa, K. Kazui, Z. Tan, Multi-scale and context-adaptive entropy model for image compression, in: IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2019, Long Beach, CA, USA, June 16-20, 2019, Computer Vision Foundation / IEEE, 2019, p. 0.
URL http://openaccess.thecvf.com/content_CVPRW_2019/html/CLIC_2019/Zhou_Multi-scale_and_Context-adaptive_Entropy_Model_for_Image_Compression_CVPRW_2019_paper.html

[25] D. Minnen, S. Singh, Channel-wise autoregressive entropy models for learned image compression, in: IEEE International Conference on Image Processing, ICIP 2020, Abu Dhabi, United Arab Emirates, October 25-28, 2020, IEEE, 2020, pp. 3339–3343. doi:10.1109/ICIP40778.2020.9190935.
URL https://doi.org/10.1109/ICIP40778.2020.9190935

[26] Y. Hu, W. Yang, J. Liu, Coarse-to-fine hyper-prior modeling for learned image compression, in: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020, AAAI Press, 2020, pp. 11013–11020.
URL https://aaai.org/ojs/index.php/AAAI/article/view/6736

[27] N. Johnston, E. Eban, A. Gordon, J. Ballé, Computationally efficient neural image compression, CoRR abs/1912.08771. arXiv:1912.08771.
URL http://arxiv.org/abs/1912.08771

[28] Y. Chen, H. Fan, B. Xu, Z. Yan, Y. Kalantidis, M. Rohrbach, S. Yan, J. Feng, Drop an octave: Reducing spatial redundancy in convolutional neural networks with octave convolution, in: 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019, IEEE, 2019, pp. 3434–3443. doi:10.1109/ICCV.2019.00353.
URL https://doi.org/10.1109/ICCV.2019.00353

[29] M. Akbari, J. Liang, J. Han, C. Tu, Generalized octave convolutions for learned multi-frequency image compression (2020). arXiv:2002.10032.

[30] J. Lin, M. Akbari, H. Fu, Q. Zhang, S. Wang, J. Liang, D. Liu, F. Liang, G. Zhang, C. Tu, Variable-rate multi-frequency image compression using modulated generalized octave convolution, in: 2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP), 2020, pp. 1–6. doi:10.1109/MMSP48831.2020.9287082.

[31] J. Lee, S. Cho, M. Kim, An end-to-end joint learning scheme of image compression and quality enhancement with improved entropy minimization (2020). arXiv:1912.12817.

[32] C. Huang, H. Liu, T. Chen, Q. Shen, Z. Ma, Extreme image coding via multiscale autoencoders with generative adversarial optimization, in: 2019 IEEE Visual Communications and Image Processing (VCIP), 2019, pp. 1–4. doi:10.1109/VCIP47243.2019.8966059.

[33] O. Rippel, L. Bourdev, Real-time adaptive image compression, in: Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR. org, 2017, pp. 2922–2930.

[34] T. Dumas, A. Roumy, C. Guillemot, Autoencoder based image compression: Can the learning be quantization independent?, in: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018, IEEE, 2018, pp. 1188–1192. doi:10.1109/ICASSP.2018.8462263.
URL https://doi.org/10.1109/ICASSP.2018.8462263

[35] K. Sohn, H. Lee, X. Yan, Learning structured output representation using deep conditional generative models, in: C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, R. Garnett (Eds.), Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada, 2015, pp. 3483–3491.

[36] A. van den Oord, N. Kalchbrenner, L. Espeholt, K. Kavukcuoglu, O. Vinyals, A. Graves, Conditional image generation with pixelcnn decoders, in: D. D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, R. Garnett (Eds.), Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain, 2016, pp. 4790–4798.
URL http://papers.nips.cc/paper/6527-conditional-image-generation-with-pixelcnn-decoders

[37] Y. Choi, M. El-Khamy, J. Lee, Variable Rate Deep Image Compression with a Conditional Autoencoder, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 3146–3154. doi:10.1109/ICCV.2019.00324.

[38] C. Cai, L. Chen, X. Zhang, Z. Gao, Efficient variable rate image compression with multi-scale decomposition network, IEEE Trans. Circuits Syst. Video Technol. 29 (12) (2019) 3687–3700. doi:10.1109/TCSVT.2018.2880492.
URL https://doi.org/10.1109/TCSVT.2018.2880492

[39] Z. Cui, J. Wang, B. Bai, T. Guo, Y. Feng, G-VAE: a continuously variable rate deep image compression framework, CoRR abs/2003.02012. arXiv:2003.02012.
URL https://arxiv.org/abs/2003.02012

[40] Y. Yang, R. Bamler, S. Mandt, Variable-bitrate neural compression via bayesian arithmetic coding, CoRR abs/2002.08158. arXiv:2002.08158.
URL https://arxiv.org/abs/2002.08158

[41] J. Ballé, P. A. Chou, D. Minnen, S. Singh, N. Johnston, E. Agustsson, S. J. Hwang, G. Toderici, Nonlinear transform coding, IEEE J. Sel. Top. Signal Process. 15 (2) (2021) 339–353. doi:10.1109/JSTSP.2020.3034501.
URL https://doi.org/10.1109/JSTSP.2020.3034501

[42] Y. Zhang, K. Li, K. Li, B. Zhong, Y. Fu, Residual non-local attention networks for image restoration, in: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019, OpenReview.net, 2019.
URL https://openreview.net/forum?id=HkeGhoA5FX

[43] L. Zhou, Z. Sun, X. Wu, J. Wu, End-to-end optimized image compression with attention mechanism, in: IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2019, Long Beach, CA, USA, June 16-20, 2019, Computer Vision Foundation / IEEE, 2019, p. 0.

[44] T. Chen, H. Liu, Z. Ma, Q. Shen, X. Cao, Y. Wang, Neural image compression via non-local attention optimization and improved context modeling (2019). arXiv:1910.06244.

[45] H. Liu, T. Chen, P. Guo, Q. Shen, X. Cao, Y. Wang, Z. Ma, Non-local attention optimized deep image compression, CoRR abs/1904.09757. arXiv:1904.09757.
URL http://arxiv.org/abs/1904.09757

[46] F. Mentzer, E. Agustsson, M. Tschannen, R. Timofte, L. V. Gool, Conditional probability models for deep image compression, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 4394–4402. doi:10.1109/CVPR.2018.00462.

[47] M. Li, W. Zuo, S. Gu, J. You, D. Zhang, Learning content-weighted deep image compression, IEEE Transactions on Pattern Analysis and Machine Intelligence (2020) 1–1doi:10.1109/TPAMI.2020.2983926.

[48] T. van Rozendaal, G. Sautière, T. S. Cohen, Lossy compression with distortion constrained optimization (2020). arXiv:2005.04064.

[49] S. Kullback, R. A. Leibler, On information and sufficiency, Ann. Math. Statist. 22 (1) (1951) 79–86. doi:10.1214/aoms/1177729694.
URL https://doi.org/10.1214/aoms/1177729694

[50] Johannes Ballé and Sung Jin Hwang and Nick Johnston and David Minnen, Tensorflow compression, available from https://tensorflow.github.io/compression/.

[51] M. Basseville, I. V. Nikiforov, et al., Detection of abrupt changes: theory and application, Vol. 104, prentice Hall Englewood Cliffs, 1993.

[52] W. Han, Y. Yang, Statistical inference in mean-field variational bayes, arXiv preprint arXiv:1911.01525.

[53] Dataset of the CVPR workshop and challenge on learned image compression (CLIC), available from http://www.compression.cc.

[54] E. Agustsson, R. Timofte, NTIRE 2017 Challenge on Single Image Super-Resolution: Dataset and Study, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2017, DIV2K dataset: DIVerse 2K resolution high quality images as used for the challenges at NTIRE (CVPR 2017 and CVPR 2018) and at PIRM (ECCV 2018), Available from https://data.vision.ee.ethz.ch/cvl/DIV2K/.

[55] H. Nemoto, P. Hanhart, P. Korshunov, T. Ebrahimi, UltraEye: UHD and HD Images Eye Tracking Dataset, in: Sixth International Workshop on Quality of Multimedia Experience (QoMEX), Singapore, 2014, available from http://mmspg.epfl.ch/ultra-eye.
URL http://infoscience.epfl.ch/record/200190

[56] L. Jin, J. Y. Lin, S. Hu, H. Wang, P. Wang, I. Katsavounidis, A. Aaron, C.-C. J. Kuo, MCL-JCI Dataset, available from http://mcl.usc.edu/mcl-jci-dataset/.

[57] Lina Jin, J. Y. Lin, S. Hu, H. Wang, P. Wang, I. Katsavounidis, A. Aaron, C.-C. J. Kuo, Statistical Study on Perceived JPEG Image Quality via MCL-JCI Dataset Construction and Analysis, in: Electronic Imaging (2016), the Society for Imaging Science and Technology (IS&T), 2016.

[58] B. Lim, S. Son, H. Kim, S. Nah, K. M. Lee, Enhanced Deep Residual Networks for Single Image Super-Resolution, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2017.

[59] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016, IEEE Computer Society, 2016, pp. 770–778. doi:10.1109/CVPR.2016.90.
URL https://doi.org/10.1109/CVPR.2016.90

[60] U. Sara, M. Akter, M. S. Uddin, Image quality assessment through fsim, ssim, mse and psnr—a comparative study, Journal of Computer and Communications 7 (3) (2019) 8–18.

[61] Z. Wang, E. P. Simoncelli, A. C. Bovik, Multiscale structural similarity for image quality assessment, in: The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003, Vol. 2, Ieee, 2003, pp. 1398–1402.

[62] R. Rassool, Vmaf reproducibility: Validating a perceptual practical video quality metric, in: 2017 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), 2017, pp. 1–2. doi:10.1109/BMSB.2017.7986143.

[63] M. H. Hyun, B. Lee, M. Kim, A frame-level constant bit-rate control using recursive bayesian estimation for versatile video coding, IEEE Access 8 (2020) 227255–227269. doi:10.1109/ACCESS.2020.3046043.

[64] T. Chen, Z. Ma, Variable bitrate image compression with quality scaling factors, in: International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2020. doi:10.1109/ICASSP40776.2020.9053885.

[65] Fei Yang and Luis Herranz and Joost van de Weijer and José A. Iglesias Guitián and Antonio M. López and Mikhail G. Mozerov, Variable rate deep image compression with modulated autoencoder, IEEE Signal Processing Letters 27 (2020) 331–335. doi:10.1109/LSP.2020.2970539.

Answers to reviewers

Paper title: Rate-Constrained Learning-based Image Compression

This manuscript is a significantly revised version four first submission for publication at Signal Processing: Image Communication.

We would like to thank the associate editor and the reviewers for their detailed reading of our manuscript and their insightful comments, which significantly helped to improve the quality of the manuscript. Extensive modifications have been performed according to the comments of the reviewers as described in the following pages. We tried our best to address every comment for both reviewers.

We believe that the proposed rate-constrained image coding solution brings a novel and refreshing tool to the learning-based image coding landscape which may still be improved with additional research investment.

The concerns from the reviewers are answered below, we also mention the modifications made in the paper and indicate where they can be found to be as clear as possible. These modifications are highlighted in RED in the paper as well.

Reviewer #1

Comment: The authors are presenting an interesting work where are proposing a non-constrained solution for solving the constrained problem of training a learning-based image codec for a specific bitrate. The paper is well written and presented; however, some more works and/or clarification are needed to support the model and the results presented in this work.

1 - In the introduction, one of the main aims of the proposed work is, as stated in the paper, "We also propose a set of heuristics to determine the new parameters introduced by our modification, without the need of intensive training for empirical determination." However, any further analysis in this direction is not given, or at least it is not spelled out. It is true that Figure 3 may supports partially the statement, but the reviewer feels that a more clear example should be added, i.e., times, number of training of proposed solution vs. the baseline.

> **Answer**: We thanks the reviewer for the comment. Moreover, we agree that few information was given about the heuristics, besides the obtained values of the loss hyper-parameters and the motivation due to the rate-shift problem. We have modified the last paragraph of the introduction. Moreover, we have included a whole section (see Section 6.3) where we discuss the complexity what you ask in your comment 3, and with it we discuss the iterations needed to specify the hyper-parameters of the loss. We clarify that around 220.000 iterations are needed to set the pair R_t, β. Once those hyper-parameters are set, the networks is completely retrained for 500.000 iterations as the baseline models. We also modified line 523 to 527 in the result sections to make this clearer.

2 - It is not clear why the authors move from eq 1 to 2 and then your model is following eq 1 (see eq. 6). A clear explanation should be given. Also it looks that parameter beta as the same scope of parameter lambda. If this is the case will be good to adopt a consistent parameters naming throughout the paper.

Answer: After revising the paper we agree that this part was a little confusing, thanks for bringing that to our attention. Actually, we have a classical RD Lagrangian formulation for rate-distortion optimization, adopted in classical encoders, which uses $J = D + \lambda R$. The variational approach, adopted in the baselines, leads to a loss function that has a similar mathematical formulation than the RD optimization in classical image encoders. However, the original work of these baselines architectures uses the λ multiplier term attached to the distortion D, leading to an equation $J = \lambda D + R$ (obviously, which term is multiplied by λ is not relevant). It is important to observe that a conventional image encoder performs RD optimization for each image. While a learning based method does not, since it is based on the trained data. Therefore, λ sets the relevance between rate and distortion for one specific trained model.

So, the classical formulation does really applies in problem. We were just trying to give some sort of context. But it was indeed confusing. Therefore, we decided to remove it, and we only maintain the Equation 1. This Lagrangian is the same obtained when we look to the Equation 5 in the baseline description. Note, that is what reviewer 2 also asked to summarize that section in the paper.

Regarding the different nomenclature λ and β, we choose to adopt different constants because the parameter which actually resembles more the λ parameter the Balle's original RD formulation is the R_t, not β. We actually discuss this similarity in the Section 6.2. In the two last paragraphs of that section we talk about the role of λ in RD formulation and the same behavior of R_t in our loss. But we decided by a term R_t because it is more attached to the interpretation of the parameter in the context of the current work. The β parameter controls the variation of the rate reconstructions around R_t. In fact, the trade-off of both parameters $\{R_t, \beta\}$ that has the full role of λ in the original formulation. Furthermore, we use this λ notation to describe models which produce many points in the rate line, while the R_t notation describe models which produce many points around a target point in the rate line. So actually, there's a different interpretation for these in different contexts.

Finally, it is important to note that we do not actually fit our loss in the form of Equation 5. Not even in the general loss where it comes from, depicted by $ELBO(\phi) = \mathbb{E}_q[\log p(\mathbf{x}|\mathbf{y})] - KL[q(\mathbf{y};\phi)||p(\mathbf{y})]$. As we now state in the paragraph of line 650, our squared term makes it hard to match a normalizable density which would lead to the proposed loss. Therefore, this loss does not come from an $ELBO$ formulation, which also would differentiate the parameters from λ. We leave it open at this work, as we continue studying it to see if we can interpret it formally in terms of a statistical inference problem.

Lastly, we also decided to simplify the text removing equations aligned to the general comment of the Reviewer #2 about the exceedingly focus of the relation of the variational mathematics to our loss, which is not our main focus on this stage of this research.

3 - The two models used are the parametric and non-parametric methods for the same architectural model. It is clear why, but it will be good to have also an example of a different architecture to see if the work proposed is working also in this case.

Answer: We appreciate the comment and we agree that further experiments with target rate using different operations in the transforms could enforce the generality of our rate loss. Therefore, we have have included the

Section 6.1 depicting an experiment with transforms whose layers are residual layers with ReLU activation in the non-parametric model. In this case, we only replaced the Conv + GDN layers by the Residual + ReLU block layers. We performed a small set of epxeriments, in the same training setup of the GND-based transforms. In the Table 2 we show the rate behavior with some statistics for the models in both validation datasets.

We have not performed the full heuristics and optimization for this model, as explained in Section 6.1, since we only wanted to show the convergence of the mean rate, in order to show that our model is useful. This is an evidence of the extension of the approach to other types of transforms. We indicated the mean PSNR in the table just to show its monotonically behavior. But optimizing the heuristic for this architecture does goes beyond the scope of this initial work. Nevertheless, this results for the Residual architecture indeed helps to improve our work. We have not tunned the β parameter for this architecture, and the standard deviation values show that the β could be smaller, which would possibly increase the mean PSNR. Also, this type of architecture might require a different training setup, like learning rate schedules, batch size, and so on, for a good generalization. We have not devised the experiment for the fine tuning of the architecture, only for the analysis of the rate behavior.

4 - Clearly spell out that JPEG AI and Kodak datasets are used for the final evaluation and comparison of the work proposed in this paper.

Answer: We thanks for the comment. We now stated this clearly in the abstract itself and also on the introduction section. As we have mentioned it in the paragraph right before the Section 6.1.

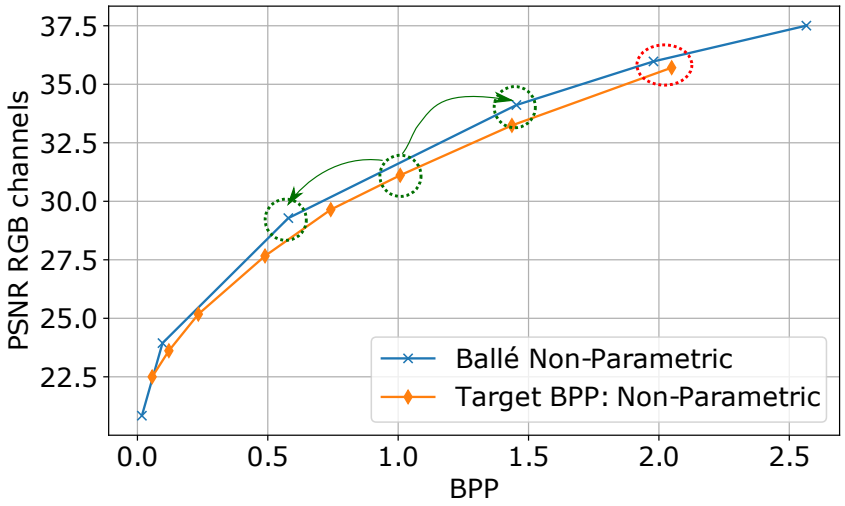
5 - The authors are talking of 500K iteration. It will be good to present a computational performances and also comparison in term of iterations for reaching the converging status of the work presented in the paper in comparison with the baseline models used in this work.

Answer: In fact, after revising the paper we agree that this was not explained as carefully as needed. Thanks to bring this to our attention, not only because the proposed heuristic is one of the contributions of this work (in order to be useful in practical real scenarios) but also to be sure that our work is reproducible.

We introduced the Section 6.3 which details more about the added complexity of the proposed heuristics and states the maximum and minimum number of iterations required in our tests. We also highlight now that once its parameters are estimated, a fresh model is trained for 500.000 iterations. This number is a reasonable number of iterations for the convergence, in terms of RD performance, to the baseline transform architectures. Therefore we have adopted it as a reference parameter. The pair $\{R_t, /beta\}$ found in the heuristics will make the network work and without the proposed heuristics, these parameters would have to be set empirically requiring several full training cycles of around 500.000 iterations.

6 - In table 1 the reviewer believe that all the images should be used for calculating the metrics, even do if few are not reaching the target bit rate, or at least it needs to be motivated why not to use all of the images.

Answer: Actually, the objective metrics of all images from the baselines architectures are not comparable. As we will explained bellow, however if this was not clearly stated in the paper, then it certainly has too. First, note that we are not trying to make it easier for our model. In fact for each line (that corresponds to a target rate) in the tables, we select the best result for all the different trained baseline models and compare it to ONE trained model from our proposal. This is key, since one of the issues with this type of learning-based image codecs is that they require several models to have similar rates between different images.



To better explain this issue lets observe the figure right above. The orange point surrounded by a green-dashed circle, which represents a reconstruction of an image at rate 1.0, cannot be fairly compared with the two closest blue dots. If we choose it to compare it with the blue dot on the left, it would be unfair because naturally with higher rates we would have lower distortion. The same logic would apply if we compare the orange dot with the blue dot to the right. Naturally the distortion measure of this point is lower as it has higher rate. However, we agree that it can be confusing since there is a lot of variables involved and we did not describe the experiments (and their motivations) good enough, .

The fair comparisons, if we are to perform it point to point, are like the one of the red dashed circle. We have near points considering the rate line, up to a threshold of distance (related to our tolerance interval), and a comparison of the distortion here is fair.

We cannot even calculate means of distortion and rates for different points in the RD plane which are far apart in the rate line. The only way compare the full set of points cannot be achievable through mean calculation like the ones show in the tables of comparison, but using the Bjøntegaard model. But the objective of our results is to show the relative deterioration of our restricted model with ones which have no rate restrictions on the loss. That's why we have chose to make fair comparisons point to point.

7 - The reviewer suggest to add a proper subjective study in section 6.4 to strength and better support the findings.

Answer: Indeed making a appropriate subjective analysis will be better and it will bring more validation to our work. However, it is very time consuming in order to follow the correct ITU recommendations. However,

we have included in the Figures used for subjective analysis (Figs. 11 - 17), the SSIM, MSIM ans VMAF metrics. All these metrics are considered have high correlation with the human visual system. Also, in these examples, best and worst cases, comparing our results to the baseline according to those metrics, are shown. Therefore, we are not selecting just our best results, we are showing best, worst for different rates. Section 6.5 was modified to better explain this in the paper.

Reviewer #2

Comment: The paper presents a modification to the objective of stochastic rate-distortion optimization in the context of neural image compression. This modification encourages the trained network to achieve a desired predetermined target bitrate.

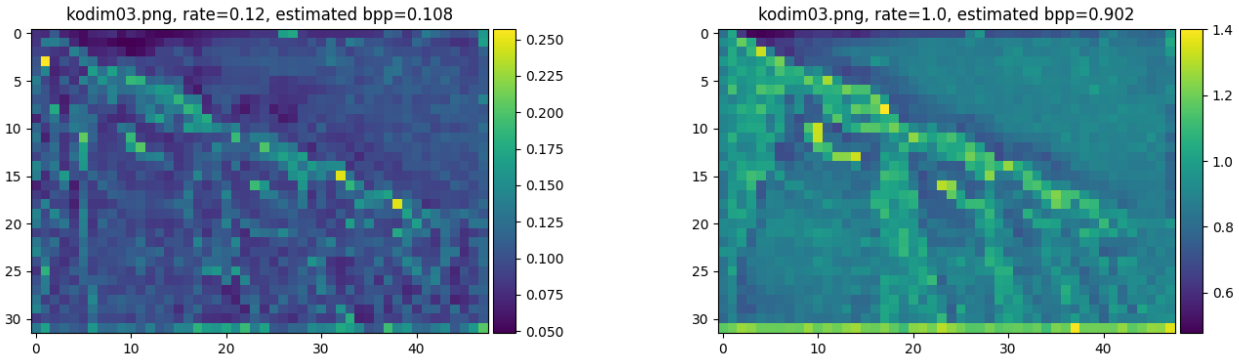
The subject of the paper is an important and practical problem: rate control for neural image compression. The paper is well organized, and appears technically sound. Overall, there is a bit too high of an emphasis on the VAE interpretation of the objective, which does not seem immediately relevant here. In my opinion, these sections of the paper could be shortened.

Answer: The authors appreciate the comment about the VAE interpretation focus and agree that exceeding emphasis was given to this aspect which is not an explored feature of the proposal in the current work. Therefore we shortened the formal explanations about the evidence lower bound derivation from Jensen's inequality (see Section 3 and Section 6.2). We also removed the initial comparison with RD otimization as suggested by Reviewer #1. Regarding the VAE relation, We have maintained most of the text in the Section 6.2, in the results section, because its main objective is to match the R_t, β parameters with the λ parameter of Balle's approach. So we maintained only the empirical view of the parameters.

Lastly, we have performed minor changes in the text to make it concise with the removals, like in the paragraph in the lines range $[210 - 213]$, where we define what a VAE is. We also make small adaptations due to the removals in the interval of lines $[650 - 661]$. In the earlier version we relied on terms which are not defined in the article anymore.

1 - My main criticism of the paper is that the authors seem to miss a crucial detail: the influence of both the "receptive field size", i.e. the size of the image patch that a single output neuron of the encoder can "see", as well as the size of the image patches used for training. The objective not only encourages each image to be encoded to a target bit rate, but also, since this is what is available during training, each individual image patch. Optimal rate allocation for a large image requires allocating more bits to complex image areas (for instance, textures and object boundaries), and fewer bits to simple areas (smooth gradients, blank regions). The proposed objective ignores this, and pushes the model towards spatially constant bit rate allocation, since the employed networks are spatially shift-equivariant. This would explain the losses of compression efficiency compared to the baseline model for some images, in particular the ones that have spatially inhomogeneous content. It would also explain its inferior performance on the parametric architecture, since this architecture is specifically designed to acknowledge the spatial variation of image content.

Answer: The authors appreciate the comments and the raised concerns. The size of receptive field certainly has a influence on the learned coding model since the image content (structural and texture patterns) within the extent of the receptive field will change. For instance, if not properly designed, the network may end up being exposed to only a small image context which may prevent it from fully exploiting spatial redundancy. In addition to taking advantage of spatial redundancy, the design of a network architecture (setting up the number of layers, stride, filter size, among other parameters) has to consider parameter count and computational burden. The present work builds upon proven model architectures, exhaustively tested, therefore benefiting from their design effort, trade-offs, and optimization. Regarding spatial bitrate allocation, the reviewer has a good point. In order to meet the target rate, the proposed loss function penalizes deviation from the target rate and treats deviations below and above the target equally. As pointed out by the reviewer, for images exhibiting smooth inhomogeneous content, there is a noticeable performance drop. This is due to the lack of a tool to move over the rate-distortion trade-off curve when bit budget is available. Most deep learning-based image coding solutions train a set of models to achieve a few rate-distortion operational points and recent works have attempted to fill the gap [37, 64, 65], however without a tool to meet a target bitrate. The figure below shows the spatial bit allocation map at 0.12 bpp and 1.0 bpp for *Kodim03*, evidencing that the network preserves its spatial bit allocation capability. The authors have created the Section 6.6 to discuss the points raised by the reviewer.



(a) 0.12 bpp

(b) 1.0 bpp

Spatial bit allocation map

2 - The paper is missing a reference to "Nonlinear Transform Coding" by Balle et al, as well as "Autoencoder Based Image Compression: Can the Learning be Quantization Independent?" by Dumas et al. Both present lambda-conditional approaches to training models for varying target bitrates. The latter is the first study in the literature proposing this that I am aware of.

Answer: The authors are aware of the mentioned research works. We have now included a description of the proposals on these references in the Section 2. There is a description of one of the cited works in lines $[143 - 151]$ and the other in lines $[165 - 174]$.

3 - Other than a missed opportunity to discuss the aspects of the problem relating to spatial variation of image complexity, the results of the paper appear mildly novel and legitimate.

Answer: The authors have created the Section 6.6 to discuss the points raised by the reviewer.