COOL-CHIC: Coordinate-based Low Complexity Hierarchical Image Codec

Théo Ladune, Pierrick Philippe, Félix Henry, Gordon Clare, Thomas Leguay
Orange Innovation, France
firstname.lastname@orange.com

Abstract

We introduce COOL-CHIC, a Coordinate-based Low Complexity Hierarchical Image Codec. It is a learned alternative to autoencoders with 629 parameters and 680 multiplications per decoded pixel. COOL-CHIC offers compression performance close to modern conventional MPEG codecs such as HEVC and is competitive with popular autoencoder-based systems. This method is inspired by Coordinate-based Neural Representations, where an image is represented as a learned function which maps pixel coordinates to RGB values. The parameters of the mapping function are then sent using entropy coding. At the receiver side, the compressed image is obtained by evaluating the mapping function for all pixel coordinates. COOL-CHIC implementation is made open-source[1].

1. Introduction and related work

For years, ITU/MPEG image and video compression algorithms (HEVC [1], VVC [2]) have been refining a coding scheme based on the separate optimization of hand-crafted linear operations. These conventional methods are being challenged by learned ones, relying on autoencoders to perform a non-linear mapping from the signal to a compact representation. Ballé's autoencoder with hyperprior [3] is a popular example of such architecture. These systems are optimized end-to-end on a large variety of samples. Once the training stage is completed, autoencoders can generalize to unseen data i.e. they compress all sort of images.

The recent JPEG-AI call for proposal [4] highlights that learned image codecs outperform state-of-the-art conventional codecs (VVC). Yet, the performance of these autoencoders comes at the expense of a tremendous complexity increase. Indeed, these codecs [5, 6] have millions of parameters and require up to a million multiplications to decode a single pixel. Thus, learned decoders are several orders of magnitude more complex than conventional ones, which might hinder their adoption.

[1] https://orange-opensource.github.io/Cool-Chic/

In 2021, Dupont introduced COIN [7], an image codec relying on Coordinates Neural Representation (CNR) to obtain a compact representation of an image. COIN models a signal by an overfitted Multi Layer Perceptron (MLP), performing the mapping from a pixel coordinates to its RGB value. The encoding stage consists in overfitting the MLP to reconstruct the image to code. The MLP parameters are then quantized and sent to the receiver. Finally, performing a forward pass to evaluate the RGB value at each spatial location allows the receiver to reconstruct the image. Thus, CNR-based codecs use a lightweight and overfitted decoder, unlike the complex and universal decoder of autoencoder approaches. For instance COIN decoder features a 10 000-parameter MLP, with performance close to JPEG.

One of the major limitations of early CNR-based approaches (COIN, NeRF [8]) is the *non-local* nature of the MLP. Indeed all parameters of the MLP contribute to the RGB value of all output pixels, regardless of their position, making the optimization of the parameters difficult. Recent work such as Instant-NGP [9] or NVP [10] circumvents this by complementing the MLP with a latent representation of the image, describing different spatial locations with different latent parameters.

This paper introduces COOL-CHIC, a Coordinate-based Low Complexity Hierarchical Image Codec. It supplements COIN with a hierarchical latent representation, which contains most of the information about the image. To properly handle the latent representation, we rely on an autoregressive model to compress it through entropy coding. In summary, the contributions of this work are as follows:

1. COOL-CHIC is a learned codec with 680 multiplications per decoded pixel. This is orders of magnitude less complex than autoencoders, paving the way for learned image decoding without dedicated hardware;
2. COOL-CHIC offers coding performance close to HEVC and competitive with Ballé's hyperprior-based autoencoder, significantly improving COIN results;
3. COOL-CHIC software is made open-source.

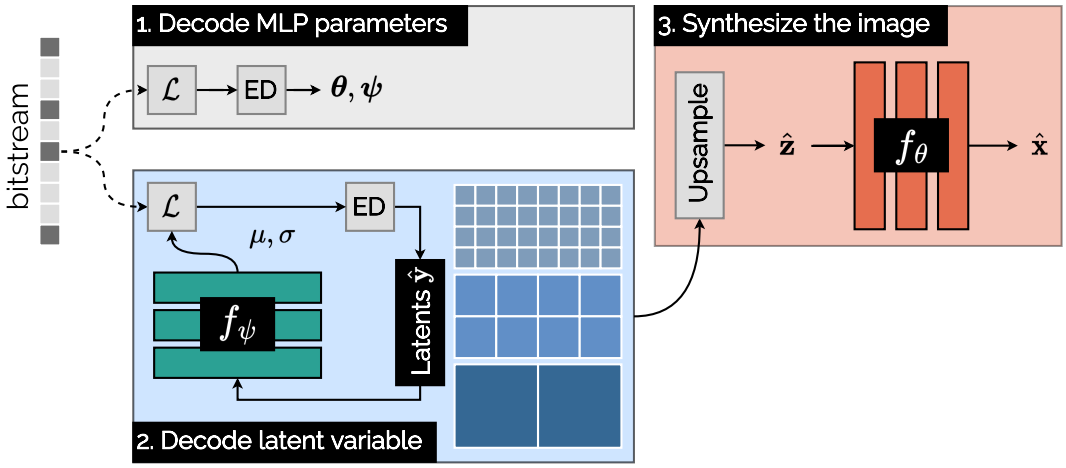


Figure 1: Decoding an image with COOL-CHIC. ED stands for entropy decoding and $\mathcal{L}$ is a Laplace distribution.

2. Proposed method

2.1. Problem statement and system overview

Let $\mathbf{x} \in \mathbb{N}^{H \times W \times F}$ be an $H \times W$ image with F color channels. Following a lossy coding setting, the original image is allowed to be distorted into $\hat{\mathbf{x}}$ to further decrease its rate. The objective of lossy coding is to minimize both the rate required to transmit $\hat{\mathbf{x}}$ and the distortion between $\mathbf{x}$ and $\hat{\mathbf{x}}$. This is stated as the minimization of the loss function:

$$\min \ \mathrm{D}\left(\mathbf{x}, \hat{\mathbf{x}}\right) + \lambda \mathrm{R}\left(\hat{\mathbf{x}}\right), \tag{1}$$

where D is a distortion metric (e.g. MSE) and R denotes the rate in bits per pixel. The rate-control parameter $\lambda \in \mathbb{R}$ balances the rate-distortion tradeoff.

Decoding an image with COOL-CHIC consists of three main steps, shown in Fig. 1. First, the parameters of two MLPs ($f_{\boldsymbol{\psi}}$ and $f_{\boldsymbol{\theta}}$) are retrieved from the bitstream. Then, $f_{\boldsymbol{\psi}}$ is used to decode $\hat{\mathbf{y}}$, a set of L 2-dimensional discrete latent variables. The latent variables are upsampled and concatenated as a dense 3D representation $\hat{\mathbf{z}} = \texttt{upsample}(\hat{\mathbf{y}})$. Finally, the RGB value of each pixel $\hat{\mathbf{x}}_{ij}$ from the compressed image is computed by feeding the dense latent representation to the synthesis MLP:

$$\hat{\mathbf{x}}_{ij} = f_{\boldsymbol{\theta}}(\hat{\mathbf{z}}_{ij}). \tag{2}$$

Encoding an image is achieved by overfitting the parameters $\{\hat{\mathbf{y}}, \boldsymbol{\psi}, \boldsymbol{\theta}\}$ so that they minimize the rate-distortion cost of the image stated in eq. (1). These parameters are learned for a single image and are therefore adapted for this content. Thus, the parameters $\{\hat{\mathbf{y}}, \boldsymbol{\psi}, \boldsymbol{\theta}\}$ are the compressed representation of the image $\mathbf{x}$. Efficient transmission of these parameters uses entropy coding, which relies on an estimate p of the signal's (unknown) probability distribution q. Asymptotically, entropy coding algorithms offer a rate close to the cross-entropy of the signal:

$$R(\hat{\mathbf{y}}) = \mathbb{E}_{\hat{\mathbf{y}} \sim q}\left[-\log_2 p(\hat{\mathbf{y}})\right]. \tag{3}$$

Here, special attention is dedicated to the latent distribution $p(\hat{\mathbf{y}})$, as its dimension is several orders of magnitude larger than that of the MLP parameters $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$. To this end, a lightweight auto-regressive probability model $p_{\boldsymbol{\psi}}\left(\hat{\mathbf{y}}\right)$ is implemented by an MLP $f_{\boldsymbol{\psi}}$. COOL-CHIC strives to minimize the RD cost expressed in eq. (1), rewritten to more clearly expose the optimized quantities:

$$\min_{\hat{\mathbf{y}}, \boldsymbol{\theta}, \boldsymbol{\psi}} \ \mathrm{D}\left(\mathbf{x}, f_{\boldsymbol{\theta}}(\texttt{upsample}(\hat{\mathbf{y}}))\right) - \lambda \log_2 p_{\boldsymbol{\psi}}\left(\hat{\mathbf{y}}\right). \tag{4}$$

This objective function does not account for the rate associated to the MLP parameters $\{\boldsymbol{\theta}, \boldsymbol{\psi}\}$, since they only contribute marginally to the overall rate. Entropy coding of the MLP parameters is achieved with a *non*-learned distribution, estimated once the optimization is complete.

In the following, Section 2.2 presents the synthesis module which reconstructs the decoded image $\hat{\mathbf{x}}$ from the latent variables. Section 2.3 introduces the auto-regressive module which models the distribution $p_{\boldsymbol{\psi}}(\hat{\mathbf{y}})$. Finally, Section 2.4 describes how the MLP parameters are transmitted.

2.2. Synthesis module

COOL-CHIC synthesis is depicted in Fig. 2. Inspired by recent CNR approaches, it relies on a latent representation $\hat{\mathbf{y}}$ fed to a lightweight MLP $f_{\boldsymbol{\theta}}$. This latent representation consists of L channels of different spatial resolutions:

$$\hat{\mathbf{y}} = \left\{\hat{\mathbf{y}}_k \in \mathbb{Z}^{H_k \times W_k}, \ k = 0, \dots, L-1\right\}, \tag{5}$$
$$\text{with } H_k = \frac{H}{2^k} \text{ and } W_k = \frac{W}{2^k}.$$

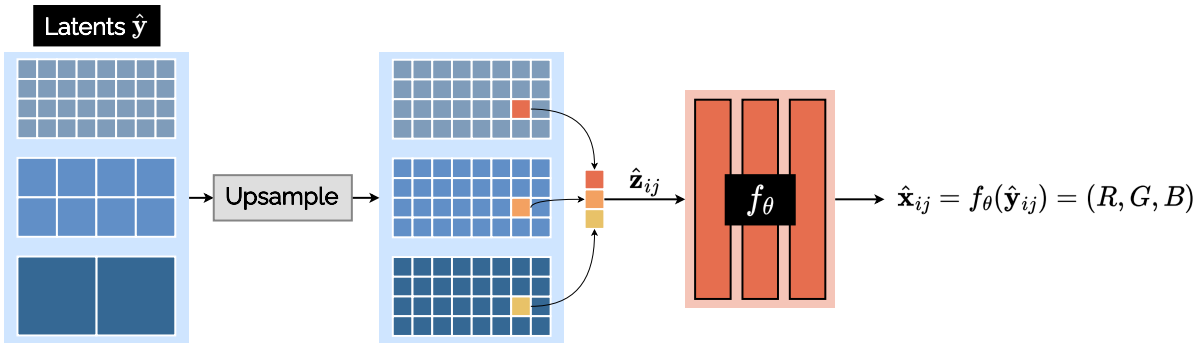


Figure 2: COOL-CHIC synthesis illustrated for a 4×8 image with $L = 3$ latent channels.

The hierarchical nature of $\hat{\mathbf{y}}$ allows for a compact representation of low-frequency areas thanks to the lowest resolutions of $\hat{\mathbf{y}}$, while still being able to capture fine details on the highest resolutions. Typically, $L = 7$ is used so that the coarsest latent resolution is $\frac{H}{64} \times \frac{W}{64}$.

Each channel $\hat{\mathbf{y}}_k$ of the latent representation is then upsampled by a factor 2^k with a bicubic interpolation. The upsampled latents are concatenated along the channel dimension to obtain $\hat{\mathbf{z}} = \texttt{upsample}(\hat{\mathbf{y}})$, a dense 3D representation of size $H \times W \times L$. Finally, the RGB value of each pixel $\hat{\mathbf{x}}_{ij}$ from the compressed image is computed by sampling $\hat{\mathbf{z}}$ at the desired position and feeding the resulting L-dimensional vector to the synthesis MLP:

$$\hat{\mathbf{x}}_{ij} = f_{\boldsymbol{\theta}}(\hat{\mathbf{z}}_{ij}), \text{ with } \hat{\mathbf{z}}_{ij} = \{\hat{z}_{ijk}, k = 0, \dots, L-1\}. \quad (6)$$

The whole synthesis module ($f_{\boldsymbol{\theta}}$ and $\hat{\mathbf{y}}$), is obtained by gradient descent to minimize the rate-distortion cost stated in eq (4). However, the latent representation is required to be discrete in order to be entropy coded. As it is not possible to directly optimize a discrete variable through a gradient descent-based optimization, a continuous latent representation $\mathbf{y}$ is learnt as a proxy. Moreover, the quantization operation is replaced by noise addition during training [11]:

$$\hat{\mathbf{y}} = \begin{cases} \mathbf{y} + \mathbf{u} & \text{with } \mathbf{u} \sim \mathcal{U}[-0.5, 0.5] \quad \text{if learning } \mathbf{y}, \\ Q(\mathbf{y}) & \text{with } Q \text{ a uniform quantizer} \quad \text{otherwise.} \end{cases} \quad (7)$$

2.3. Auto-regressive probability model

The role of the auto-regressive probability model $p_{\boldsymbol{\psi}}$ is highlighted by rewriting eq. (3):

$$\begin{aligned} R(\hat{\mathbf{y}}) &= \mathbb{E}_{\hat{\mathbf{y}} \sim q}\left[-\log_2 p(\hat{\mathbf{y}})\right] \\ &= \mathbb{E}_{\hat{\mathbf{y}} \sim q}\left[-\log_2 p_{\boldsymbol{\psi}}(\hat{\mathbf{y}}) + \log_2 q(\hat{\mathbf{y}}) - \log_2 q(\hat{\mathbf{y}})\right] \\ &= D_{KL}\left(q \,||\, p_{\boldsymbol{\psi}}\right) + \mathrm{H}(\hat{\mathbf{y}}). \end{aligned} \quad (8)$$

Here, D_{KL} stands for the Kullback-Leibler divergence and H for Shannon's entropy. Equation (8) states that it is possible to act on two terms to minimize the rate. First, decreasing the entropy (i.e. the average information quantity) of the latent variable $\hat{\mathbf{y}}$. This is at the heart of the rate-distortion tradeoff as less information in $\hat{\mathbf{y}}$ implies more distortion in $\hat{\mathbf{x}}$. The second means of reducing the rate is to estimate a distribution $p_{\boldsymbol{\psi}}$ as close as possible to the actual (unknown) latent distribution q. This is the role of the auto-regressive module shown in Fig. 3.

Modeling the joint distribution of $\hat{\mathbf{y}}$ is untractable due to its high dimension. Inspired by [12], we resort to a factorized model, where the distribution of each latent pixel $\hat{y}_{ijk}$ (i.e. the pixel at location (i, j) in the k-th latent channel) is conditioned on C spatially neighboring pixels $\mathbf{c}_{ijk} \in \mathbb{Z}^C$.

$$p_{\boldsymbol{\psi}}(\hat{\mathbf{y}}) = \prod_{i,j,k} p_{\boldsymbol{\psi}}(\hat{y}_{ijk} \mid \mathbf{c}_{ijk}). \quad (9)$$

Since the distribution $p_{\boldsymbol{\psi}}$ must be known to both the emitter and receiver, only causal (already received) context pixels can be used to estimate the distribution $p_{\boldsymbol{\psi}}$. Moreover, the context pixels are selected to introduce as little sequentiality as possible. As such, no inter latent channel dependency is leveraged, allowing to decode all L channels in parallel. Furthermore, rows can also be processed in parallel in a wavefront-like manner [13].

Following the usual practice in learned coding [3], the discrete distribution $p_{\boldsymbol{\psi}}(\hat{\mathbf{y}})$ of the quantized latent variable is actually modeled by integrating the *continuous* distribution of the non-quantized latent $g(\mathbf{y})$, modeled as a Laplace distribution. The MLP $f_{\boldsymbol{\psi}}$ learns to estimate the proper expectation and scale parameters of g, based on the context pixels. As such, the probability of a latent pixel is:

$$p_{\boldsymbol{\psi}}(\hat{y}_{ijk} \mid \mathbf{c}_{ijk}) = \int_{\hat{y}_{ijk}-0.5}^{\hat{y}_{ijk}+0.5} g(y)\mathrm{d}y, \quad (10)$$

$$\text{with } g \sim \mathcal{L}(\mu_{ijk}, \sigma_{ijk}) \text{ and } \mu_{ijk}, \sigma_{ijk} = f_{\boldsymbol{\psi}}(\mathbf{c}_{ijk}).$$

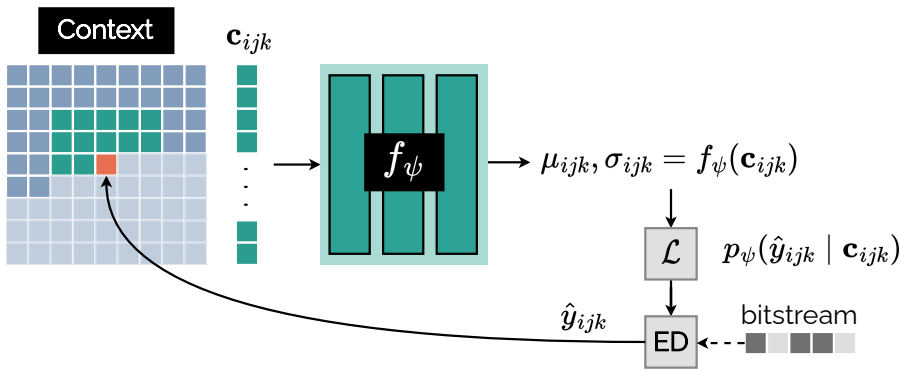


Figure 3: Entropy decoding of $\hat{y}_{ijk}$ using the auto-regressive probability model of COOL-CHIC and $C = 12$ context pixels.

Finally, the rate term present in eq. (4) sums up to:

$$\begin{aligned} R(\hat{\mathbf{y}}) &= -\log_2 p_{\boldsymbol{\psi}}(\hat{\mathbf{y}}) \\ &= -\log_2 \prod_{i,j,k} p_{\boldsymbol{\psi}}(\hat{y}_{ijk} \mid \mathbf{c}_{ijk}) \\ &= \sum_{i,j,k} -\log_2 p_{\boldsymbol{\psi}}(\hat{y}_{ijk} \mid \mathbf{c}_{ijk}). \end{aligned} \tag{11}$$

2.4. Compressing the model parameters

During the training stage (i.e. the encoding), the MLPs parameters $\{\boldsymbol{\psi}, \boldsymbol{\theta}\}$ are represented as 32-bit floating point values. Yet, they do not require such a high-precision representation once the training is finished. This section explains how the quantization accuracy is set for the MLPs.

As the synthesis $f_{\boldsymbol{\theta}}$ and the probability model $p_{\boldsymbol{\psi}}$ perform different tasks, they likely require different accuracy. Consequently, different quantization steps $\Delta_{\boldsymbol{\psi}}$ and $\Delta_{\boldsymbol{\theta}}$ are used for $\boldsymbol{\psi}$ and $\boldsymbol{\theta}$. Instead of the full-precision parameters determined by the optimization process, COOL-CHIC relies on their quantized version:

$$\begin{aligned} &\hat{\boldsymbol{\theta}} = Q(\boldsymbol{\theta}, \Delta_{\boldsymbol{\theta}}) \text{ and } \hat{\boldsymbol{\psi}} = Q(\boldsymbol{\psi}, \Delta_{\boldsymbol{\psi}}), \\ &\quad \text{with } Q(\cdot, \Delta) \text{ a scalar quantizer of step } \Delta. \end{aligned} \tag{12}$$

A probability model of $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\psi}}$ is required to send them *via* an entropy coding algorithm. Similarly to the quantized latent variable distribution, the discrete distribution of each quantized MLP parameter is modeled *via* a continuous Laplace distribution, see eq. (11). As such, the probability of one parameter from $\boldsymbol{\theta}$ (the same holds for $\boldsymbol{\psi}$) is:

$$\begin{aligned} &p(\hat{\theta}_i) = \int_{\hat{\theta}_i - 0.5}^{\hat{\theta}_i + 0.5} g(\theta) \mathrm{d}\theta, \\ &\quad \text{with } g \sim \mathcal{L}\left(0, \sigma_{\hat{\boldsymbol{\theta}}}\right) \text{ and } \sigma_{\hat{\boldsymbol{\theta}}} = \texttt{stddev}(\hat{\boldsymbol{\theta}}) \end{aligned} \tag{13}$$

The total rate contribution of both MLPs is estimated as:

$$\begin{aligned} \mathrm{R}_{\mathrm{MLP}} &= R_{\hat{\boldsymbol{\theta}}} + R_{\hat{\boldsymbol{\psi}}} \\ &= \sum_{\hat{\theta}_i} -\log_2 p(\hat{\theta}_i) + \sum_{\hat{\psi}_j} -\log_2 p(\hat{\psi}_j). \end{aligned} \tag{14}$$

Quantization of the MLP parameters allows the reduction of $\mathrm{R}_{\mathrm{MLP}}$ at the expense of the probability model and synthesis accuracy. Consequently, it is important to properly select the value of $\Delta_{\boldsymbol{\psi}}$ and $\Delta_{\boldsymbol{\theta}}$. This is achieved by a greedy minimization of the rate-distortion cost associated to different quantization steps (e.g. from 10^{-1} to 10^{-5}). For each quantization step, $\mathrm{R}_{\mathrm{MLP}}$ and the compression performance (distortion and latent rate when using the quantized MLPs) are measured. The selected quantization steps are those minimizing the following rate-distortion cost.

$$\min_{\Delta_{\boldsymbol{\psi}}, \Delta_{\boldsymbol{\theta}}} \mathrm{D}\left(\mathbf{x}, \hat{\mathbf{x}}\right) + \lambda \left(\mathrm{R}(\hat{\mathbf{y}}) + \mathrm{R}_{\mathrm{MLP}}\right). \tag{15}$$

3. Experimental results

3.1. Rate-distortion results

This section provides experimental results demonstrating the efficiency of COOL-CHIC as a low-complexity image decoder. To this end, the rate-distortion performance of COOL-CHIC is measured on the Kodak dataset [15] and the CLIC20 professional validation set [16] using the PSNR as quality metric (computed in the RGB444 domain).

Several anchors are provided to better appreciate COOL-CHIC results. A first set of anchors are conventional (i.e. non-learned) codecs such as JPEG and HEVC intra, where HEVC is tested using its test model HM operating in 444. The performance of COIN [17], a popular CNR-based codec from the literature, is provided. Finally, results of the well-known Ballé's autoencoder with hyperprior [3] (inferred using compressAI [14]) are also presented.

Prior CNR-based codecs (e.g. COIN) rely on varying the architecture of the synthesis MLP $f_{\boldsymbol{\theta}}$ to address different

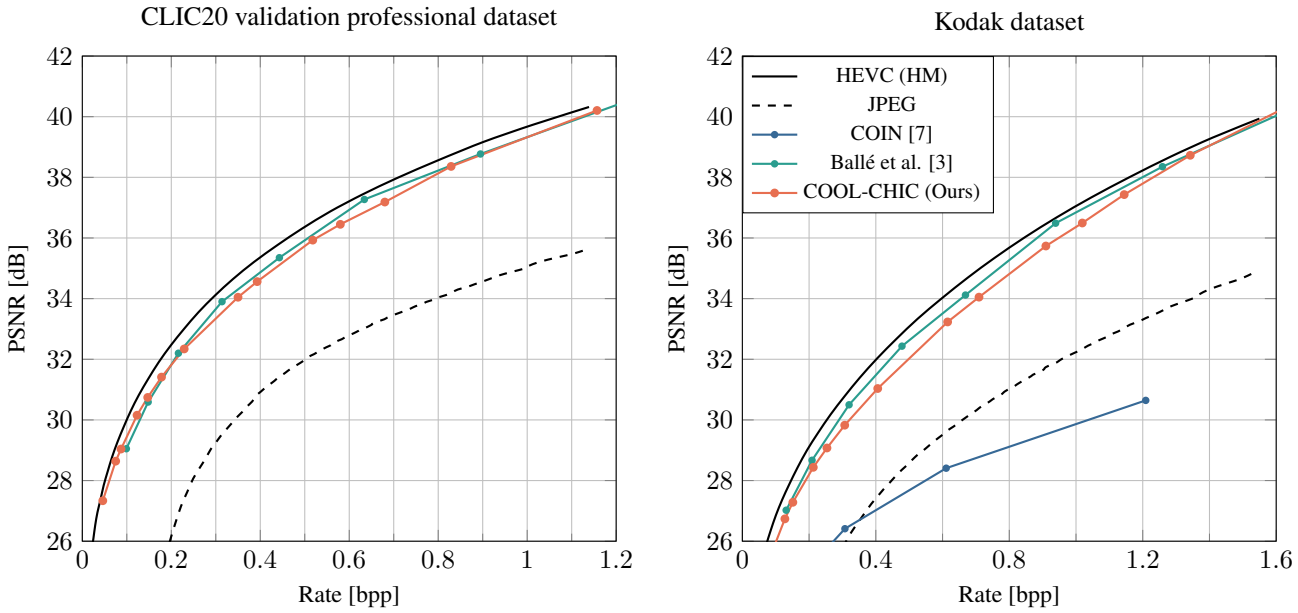


Figure 4: Rate-distortion performance on CLIC20 professional and Kodak datasets. Performance of Ballé's autoencoder come from CompressAI [14]. PSNR is computed in the RGB444 domain.

rates. Thanks to the addition of a latent representation, COOL-CHIC implements a single architecture for all rates. Both the synthesis MLP f_{θ} and the probability model MLP f_{ψ} have two hidden layers of width 12 with ReLUs. As stated in Fig. 3, $C = 12$ context pixels are leveraged and the latent variable is composed of $L = 7$ different channels. This results in a lightweight decoder, with 629 parameters and 680 multiplications per decoded pixel.

Figure 4 presents the rate-distortion results. COOL-CHIC outperforms prior CNR-based codec COIN, across the entire range of rate. Moreover, it offers performance on par with Ballé's autoencoder while offering a reduced decoding complexity. At higher rates, COOL-CHIC comes close to the performance of modern conventional codecs such as HEVC. These are compelling results since they prove that COOL-CHIC is able to compete with well-established autoencoders and conventional codecs.

It should be noted that COOL-CHIC performance appears to be better for higher rates. This is likely due to the cost of sending the MLPs for the synthesis and probability model. Figure 5 presents the evolution of the rate share for the MLPs and the latent variable. At lower rates, the rate associated to the MLPs approaches 20 % of the overall rate. This is an important overhead which might explains the worse performance of COOL-CHIC at low rates. As the MLPs rate is approximately constant for all rates, this overhead tends to decrease when the overall bitrate increases.

3.2. Complexity analysis

3.2.1 Decoder complexity

The main benefit of COOL-CHIC is the low complexity of its decoder. This is illustrated in Fig. 6a, which presents the number of kMAC (kilo multiplication-accumulation) per decoded pixel for different approaches. COOL-CHIC complexity remains consistently lower compared with prior CNR-based approaches (e.g. COIN). Furthermore, it is two orders of magnitude less complex than Ballé's

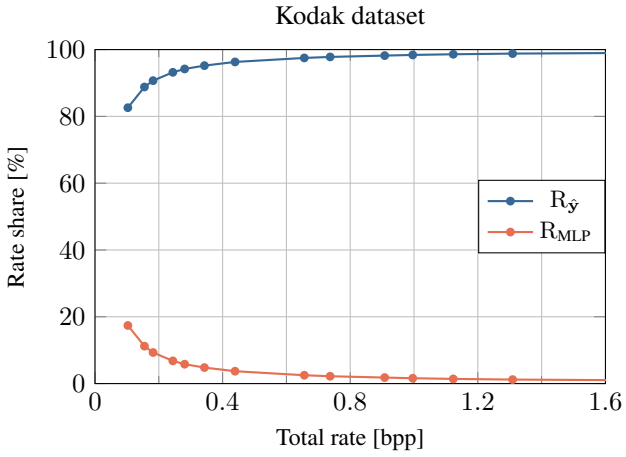
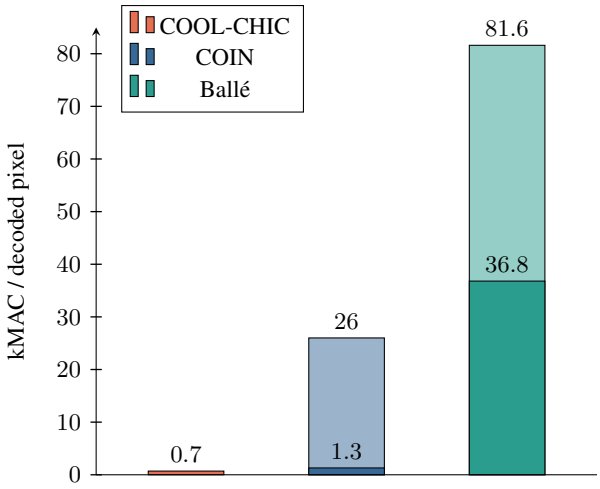
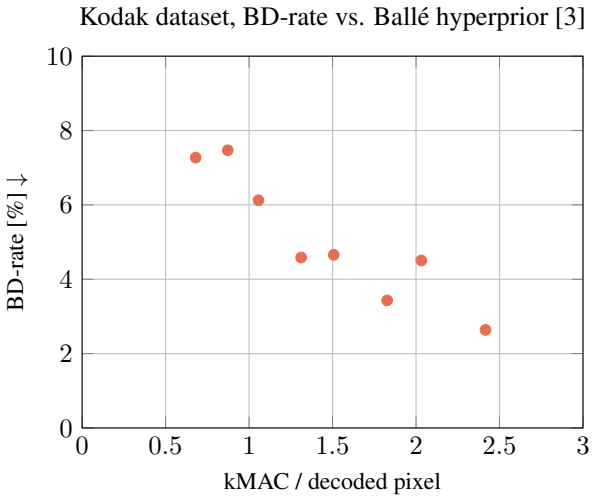


Figure 5: Contributions of the two rate terms: the MLPs rate R_{MLP} and the latent variable rate $R_{\hat{y}}$.



(a) Learned decoders minimum and maximum complexity.



(b) Performance for different decoding complexity. Lower BD-rate is better

Figure 6: Complexity analysis of COOL-CHIC decoder. kMAC stands for kilo multiplication-accumulation.

hyperprior-based autoencoder while offering competitive rate-distortion performance. This proves that COOL-CHIC is a low-complexity alternative to learned autoencoders.

While the main focus of this work is the design of a low-complexity learned decoder, it is possible to slightly increase the decoder complexity to obtain better compression performance. In particular, the number of context pixels and the width of the hidden layer are increased from 12 up to 24. Fig. 6b presents the exploration of this performance-complexity continuum. Performance is expressed as the BD-rate [18] of COOL-CHIC using Ballé's hyperprior autoencoder as reference system. BD-rate indicates the relative rate required to achieve the same quality than the reference system. The point with the smallest complexity (680 MAC / decoded pixel) corresponds to the one shown in the rate-distortion graphs of Fig. 4 and presents a BD-rate of 7 % (i.e. COOL-CHIC needs 7 % more rate to offer the same quality than Ballé's autoencoder). Increasing the complexity up to 2 500 MAC / decoded pixel leads to decrease the BD-rate to less than 3 %.

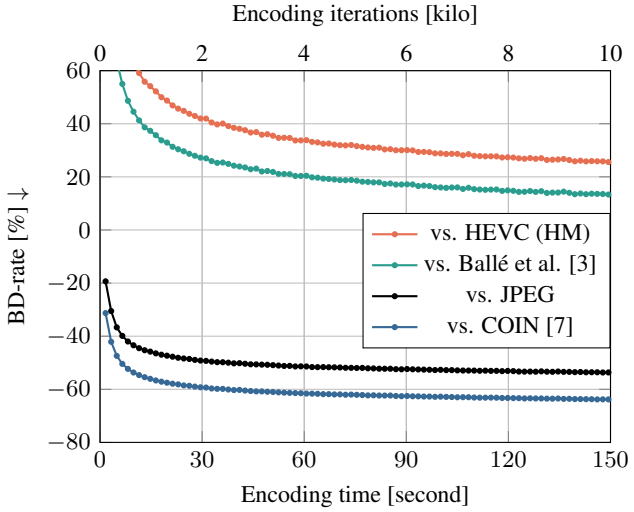
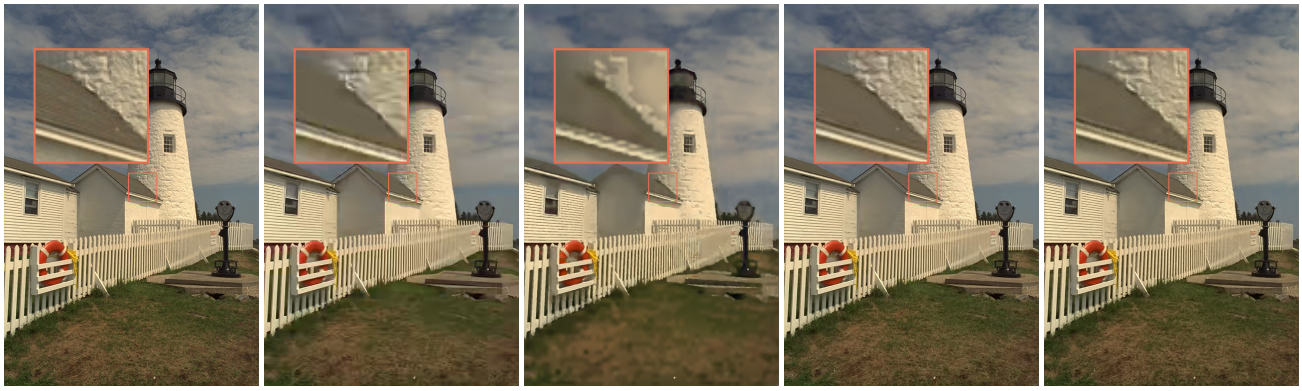


Figure 7: COOL-CHIC BD-rate on the Kodak dataset as a function of the encoding time. Negative BD-rate means that COOL-CHIC requires less rate to offer the same quality.

3.2.2 Encoder complexity

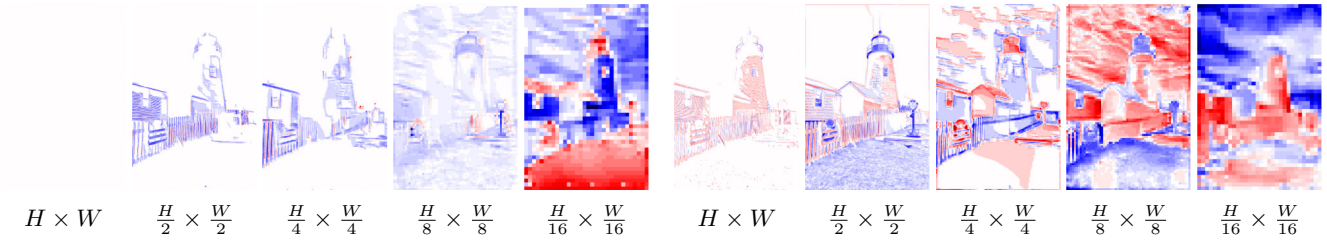
COOL-CHIC offers low decoding complexity. However, the encoding complexity is more significant since COOL-CHIC learns the latent representation, the synthesis MLP and the probability model MLP for each image. Here, the encoding of a 768×512 image takes around 10 minutes and 40 000 iterations. Yet, better implementation (e.g. CUDA/C++) would lead to a dramatic speed-up. For instance, Instant-NGP [9] learns to synthesize images (albeit without a rate constraint) in a matter of seconds. Besides reducing the duration of each iteration, meta-learning-based approaches such as COIN++ [17] offer solutions to significantly reduce the number of iterations. This hints that the encoding time of COOL-CHIC is likely not an issue.

The impact of the encoding time is evaluated by computing COOL-CHIC BD-rate against different anchors (Ballé's autoencoder, COIN, HEVC and JPEG) throughout the encoding process. The results of this experiment are



(a) Original (b) HEVC 0.142 bpp (c) Ours 0.145 bpp (d) HEVC 0.578 bpp (e) Ours 0.589 bpp

Figure 8: Comparison of HEVC and COOL-CHIC on Kodak image *kodim19* at two different rates.



(a) Rate is 0.145 bpp. (b) Rate is 0.589 bpp.

Figure 9: Latent variables for two rates (best viewed on screen). Captions indicate the resolution of each latent channel.

presented in Fig. 7. COOL-CHIC requires less than 100 iterations (1.5 second) to outperform JPEG and COIN, offering a BD-rate of -20 % (versus JPEG) and -30 % (versus COIN). After 1 minute, COOL-CHIC already offers compression performance close to Ballé's autoencoder with a BD-rate of 20 %. This results illustrates that the encoding time is not prohibitive since most of the compression efficiency is obtained in a few minutes.

Conceptually, COOL-CHIC is better compared to conventional codecs (HEVC, VVC) than to learned auto-encoders. Indeed, the encoding of COOL-CHIC consists of learning an adapted latent and transform which best suit the current image to compress. This is similar to the trial of many different coding modes in conventional codecs in order to find the best ones. It results in a high encoding complexity (often done once on a dedicated server) and a low decoding complexity (done many times on low-power devices e.g. smartphones).

3.3. Visualization

Visual examples are provided in Fig. 8, which compare HEVC and COOL-CHIC reconstruction at different rates. At low rate, both codecs exhibit significant degradations and compression artifacts. Since they operate differently, the nature of their artifacts is different. HEVC is a block-based codec and as such has blocking artifacts visible on the lighthouse wall or in the grass. COOL-CHIC presents other kind of artifacts akin to banding artifacts caused by the upscaling of low-resolution latent channels. At higher rates, both HEVC and COOL-CHIC are able to produce high-quality reconstructions without coding artifacts.

Unlike prior CNR-based codecs, COOL-CHIC does not rely on varying the MLPs architecture to address different rates. Instead, the rate-distortion tradeoff concerns mostly the latent representation, whose entropy decreases to reduce its rate. This is depicted in Fig. 9 which presents the 5 highest-resolution latent variables obtained for two different rates. The top row in Fig. 9 allows reconstruction of the image presented in Fig. 8c and the bottom row corresponds to the image in Fig. 8e. At high-rate COOL-CHIC uses all latent resolutions, including the highest one, which is not the case at low rate. Furthermore, low-rate latent channels are sparser and have smaller entropy than the high-rate ones (e.g. the $\frac{H}{2} \times \frac{W}{2}$ channel). This shows that the rate-distortion optimization of COOL-CHIC learns to populate the latent variable according to the rate constraint.

4. Conclusion and future work

This work proposes COOL-CHIC, a Coordinate-based Low Complexity Hierarchical Image Codec. It is built on top of a Coordinate-based Neural Representation (CNR), complemented by a hierarchical latent representation. By overfitting the whole system for each image to compress, COOL-CHIC is able to achieve compelling rate-distortion results for a reduced decoder-side complexity of 680 multiplications per decoded pixels. In particular, COOL-CHIC offers performance on par with Ballé's hyperprior-based autoencoder while being two orders of magnitude less complex. Moreover, COOL-CHIC performance comes close to modern conventional codecs such as HEVC. This yields promising perspectives, as the low decoder complexity paves the way for real-life usage of learned compression.

While image coding performance is already compelling, further progress is still needed to compete with state-of-the-art codecs (VVC), especially at lower rates. Also, studying how to extend COOL-CHIC to video coding is of primary interest, as video compression requires low-complexity decoding in order to ensure real-time decoding. Finally, reducing the encoding time is necessary to rely on COOL-CHIC for practical use cases.

References

[1] Gary J. Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the high efficiency video coding (HEVC) standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 2012.

[2] Benjamin Bross, Jianle Chen, Jens-Rainer Ohm, Gary J. Sullivan, and Ye-Kui Wang. Developments in international video coding standardization after AVC, with an overview of versatile video coding (VVC). *Proceedings of the IEEE*, 2021.

[3] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[4] ISO/IEC JTC 1/SC29/WG1 N100250, REQ "report on the JPEG AI call for proposals results", 2022.

[5] Y. Ma, Y. Zhai, W. Jiang, I. Li, Z. Yang, and R. Wang. ROI image codec optimized for visual quality. 2022.

[6] Dailan He, Ziming Yang, Weikun Peng, Rui Ma, Hongwei Qin, and Yan Wang. ELIC: efficient learned image compression with unevenly grouped space-channel contextual adaptive coding. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 5708–5717. IEEE, 2022.

[7] Emilien Dupont, Adam Golinski, Milad Alizadeh, Yee Whye Teh, and Arnaud Doucet. COIN: compression with implicit neural representations. *CoRR*, abs/2103.03123, 2021.

[8] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[9] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022.

[10] Subin Kim, Sihyun Yu, Jaeho Lee, and Jinwoo Shin. Scalable neural video representations with learnable positional features. *CoRR*, abs/2210.06823, 2022.

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

[12] David Minnen, Johannes Ballé, and George Toderici. Joint autoregressive and hierarchical priors for learned image compression. *CoRR*, abs/1809.02736, 2018.

[13] Chi Ching Chi, Mauricio Alvarez-Mesa, Ben Juurlink, Gordon Clare, Félix Henry, Stéphane Pateux, and Thomas Schierl. Parallel scalability and efficiency of HEVC parallelization approaches. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12):1827–1838, 2012.

[14] Jean Bégaint, Fabien Racapé, Simon Feltman, and Akshay Pushparaja. CompressAI: a PyTorch library and evaluation platform for end-to-end compression research. *arXiv preprint arXiv:2011.03029*, 2020.

[15] Kodak image dataset. *http://r0k.us/graphics/kodak/*.

[16] Challenge on learned image coding 2020. *http://clic.compression.cc/2021/tasks/index.html*.

[17] Emilien Dupont, Hrushikesh Loya, Milad Alizadeh, Adam Golinski, Yee Whye Teh, and Arnaud Doucet. COIN++: data agnostic neural compression. *CoRR*, abs/2201.12904, 2022.

[18] Gisle Bjontegaard. Calculation of average psnr differences between rd-curves. *VCEG-M33*, 2021.