Joint Global and Local Hierarchical Priors for Learned Image Compression

Jun-Hyuk Kim1*  Byeongho Heo2  Jong-Seok Lee1
1School of Integrated Technology, Yonsei University  2NAVER AI Lab

Abstract

Recently, learned image compression methods have shown superior performance compared to the traditional hand-crafted image codecs including BPG. One of the fundamental research directions in learned image compression is to develop entropy models that accurately estimate the probability distribution of the quantized latent representation. Like other vision tasks, most of the recent learned entropy models are based on convolutional neural networks (CNNs). However, CNNs have a limitation in modeling dependencies between distant regions due to their nature of local connectivity, which can be a significant bottleneck in image compression where reducing spatial redundancy is a key point. To address this issue, we propose a novel entropy model called Information Transformer (Informer) that exploits both local and global information in a content-dependent manner using an attention mechanism. Our experiments demonstrate that Informer improves rate-distortion performance over the state-of-the-art methods on the Kodak and Tecnick datasets without the quadratic computational complexity problem.

1. Introduction

More than one trillion photos are taken every year and the number is increasing [12], leading to an ever-increasing demand for improved compression efficiency. Recently, advances in deep learning have led to significant progress in learned image compression [7, 8, 16, 25, 28, 32, 37, 41–43]. Generally, learned image compression follows a transform coding framework [21] consisting of transformation and entropy coding (see the top of Fig. 1). In this framework, an image is first transformed into a quantized latent representation that enables more effective compression than the original image. Then, the quantized latent representation is encoded to a bitstream by a standard entropy coding algorithm (e.g., arithmetic coding [40]). An entropy model, i.e., a prior probability model on the quantized latent representation, is required for the entropy coding algorithm. Deep neural networks are employed for the transformation and entropy model in this framework [7, 8, 16, 28, 32, 37, 41], where both are learned in an end-to-end manner to fully utilize the strong capability of deep neural networks [24]. Since the length of the bitstream relies on the entropy model, designing an accurate entropy model is important for compression efficiency, which is our main focus in this paper. The goal of entropy models is to estimate a joint probability distribution over the elements of the quantized latent representation. A simple way to do this is to assume complete independence among the elements [7]. However, this approach yields limited compression efficiency [8]. Thus, how to model the dependencies between the elements of the quantized latent representation has been an important issue in learned image compression [8, 28, 32, 37]. It is popular to extract additional features, called “hyperprior” or “hierarchical prior”, capturing the dependencies from the latent representation using convolutional neural networks (CNNs). This approach has contributed to learning accurate entropy models, making learned image compression methods outperform hand-crafted image codes such as BPG [10]. However, despite the significant progress, CNN-based entropy models still have limitations in capturing the dependencies due to the nature of CNNs. First, existing entropy models do not make full use of global information due to the local receptive field of CNNs. This issue can be critical in modeling long-range dependencies. For exam-
ple, in the case of Fig. 1, the CNN-based approach cannot capture well the red windows that repeatedly appear across the whole image due to the localized receptive field. Second, the receptive fields of previous entropy models cannot exclude nearby elements with different contents due to the content-independent property of convolution operations [34]. In other words, no matter how different the contents of two elements are, they are processed within the same receptive field if they are located nearby. In Fig. 1, although the red window and the bricks have quite different contents, both are used simultaneously in the process of capturing dependencies. These issues can limit the performance of learned entropy models.

To overcome these issues, we propose a novel entropy model, called Information Transformer (Informer), that captures both global and local dependencies in a content-dependent manner using the attention mechanism of Transformer [44] (Fig. 1). In contrast to convolution operations, the attention mechanism has known to be effective in modeling long-range dependencies in a content-dependent manner [34]. Based on the joint autoregressive and hierarchical priors [32], which is the basis of the latest entropy models [16, 37], we introduce two novel hyperpriors, i.e., a global hyperprior and a local hyperprior. To model global dependencies of the quantized latent representation, our Informer first extracts a global hyperprior consisting of different vectors that attend to different areas of an image by using the cross-attention mechanism [4, 14, 30]. Furthermore, our Informer extracts a local hyperprior specialized for local information by using $1 \times 1$ convolutional layers. Our local hyperprior prevents the global hyperprior from utilizing only local information and thus allows our Informer to consider local and global information effectively.

Compared to the baseline model [32], Informer improves the rate–distortion performance of learned image compression methods on the popular Kodak [27] and Tecnick [2] datasets. In addition, Informer achieves better performance than the recently proposed global reference model [37] aiming at capturing global dependencies; Informer not only yields higher rate–distortion performance but also avoids the quadratic computational complexity problem of the global reference model [37]. Our main contributions can be summarized as follows:

- We propose joint global and local hyperpriors that effectively model two different types of dependencies between the elements of the quantized latent representation using the attention mechanism.
- We demonstrate that our Informer with the joint global and local hyperpriors improves rate–distortion performance of learned image compression while addressing the quadratic computational complexity problem.

2. Related work

Learned nonlinear transforms. One of the keys to the success of learned image compression is that deep neural networks effectively model nonlinear transforms suitable for image compression, while traditional image codecs mostly assume linear transforms due to the difficulty of hand-engineering nonlinear transforms for high-dimensional data like images [5]. Since Ballé et al. [6] proposed the generalized divisive normalization (GDN) layer that is effective for modeling nonlinear transforms, CNNs with the GDN layers have been widely used in later methods [7, 8, 28, 32]. Recently, some learned image compression methods have been proposed using deep residual networks with small kernels (i.e., $3 \times 3$) [15], an attention module [16], invertible neural networks [48], and an attentional multi-scale back-projection module [20].

Attention mechanisms. The attention mechanism [44] is one of the most successful methods to handle global information in deep neural networks. It demonstrates notable performance in the language domain through the Transformer architecture [44]. Some researches [46, 47] tried to utilize the powerful performance of the attention mechanism in the computer vision domain. Recently, Vision Transformer [18] achieves state-of-the-art accuracy on image classification tasks. Many studies have been conducted to use and improve Vision Transformer in diverse vision tasks such as image classification [18, 23], object detection [13, 29], and semantic segmentation [29, 50]. Since Transformer has a strong ability to model long-range dependencies regardless of their distance in the pixel domain [39], which existing learned entropy models do not have, we propose a novel Transformer-based learned entropy model.

3. Joint global and local hyperpriors

3.1. Learned image compression

Given the input image $x$, most learned image compression models [8, 16, 32] aim to jointly minimize the expected length of the bitstream (i.e., rate) and the expected distortion of the decoded image with respect to $x$:

$$
\min_{\hat{y}} \mathbb{E}_{x \sim p_x} [\log_2 p_{\hat{y}}(\hat{y}(x))] + \lambda \cdot \mathbb{E}_{x \sim p_x} [d(x, \hat{y}(\hat{y}(x)))]
$$

\[ \lambda \] is the Lagrange multiplier controlling the trade-off between rate and distortion. $f_a(\cdot)$, $\cdot$, and $f_s(\cdot)$ represent an encoder, a quantizer, and a decoder, respectively. $\hat{y}$ is the quantized latent representation, i.e., $\hat{y} = f_a(x)$. $p_x$ is the distribution of training images, and $p_{\hat{y}}$ is the learned entropy model. When an entropy coding proceeds under the learned entropy model $p_{\hat{y}}$, the smallest rate is the cross-entropy between the actual distribution of the quantized latent representation and the learned entropy model $p_{\hat{y}}$, which is thus
3.2. Learned entropy models

The entropy models seek to estimate a joint probability distribution over the elements of the quantized latent representation $\hat{y}$. Note that the rate term in Eq. (1) is minimized when the learned entropy model $p_\theta$ perfectly matches the actual distribution. A simple approach to model the distribution of $\hat{y}$ is to assume that all elements are statistically independent and to learn a fixed entropy model, i.e., fully factorized model [7, 41]. Despite its simplicity, this approach does not model the remaining dependencies in $\hat{y}$, and thus cannot achieve optimal performance [8].

To address this limitation, advanced methods [8, 28, 32] propose conditional entropy models where the elements are assumed to follow conditionally independent parametric probability models, and the distribution parameters are adapted by utilizing the remaining dependencies. They can be divided into two directions: what parametric models to be used [8, 16, 17, 32] and how to model dependencies [8, 28, 32, 37]. The former direction includes zero-mean Gaussian [8], Gaussian [32], Gaussian mixture [16], and asymmetric Gaussian [17]. Among them, we employ the widely used one, i.e., Gaussian [32]. Specifically, we use a Gaussian distribution convolved with a unit uniform distribution following the previous works [8, 32]:

$$p_\theta(\hat{y}) = \prod_i \mathcal{N}(\mu_i, \sigma_i^2) \ast \mathcal{U}(-\frac{1}{2}, \frac{1}{2})(\hat{y}_i),$$

where $\mu_i$ and $\sigma_i$ are the mean and scale parameters for each element $\hat{y}_i$, respectively.

The main focus of this study is on the latter direction, i.e., accurate modeling of dependencies. Existing methods model local dependencies in two different ways. First, Ballé et al. [8] capture the local dependencies by extracting side information that is encoded additionally, which is called a hyperprior. The operational diagram of this approach is explained in Fig. 2a, where the hyperprior model $(h_a)$ extracts and utilizes the hyperprior $\hat{z}$ for predicting the distribution parameter $\sigma$. Since additional information is encoded, the rate term in Eq. (1) is extended as follows:

$$\mathbb{E}_{x \sim p_x} \left[ -\log_2 p_\theta(\hat{y}) - \log_2 p_{\hat{z}}(\hat{z}) \right].$$

where $p_{\hat{z}}$ is the learned entropy model for the hyperprior $\hat{z}$ and designed by using the non-parametric fully factorized entropy model [7].

Another approach of modeling local dependencies is to utilize previously decoded adjacent elements (i.e., a context prior [28, 32]). While the hyperprior requires additional bits, the context prior is bit-free. Since Minnen et al. [32] and Lee et al. [28] demonstrated that the two kinds of priors are complementary, they have been typically used jointly in
literature [16, 17, 20] (Fig. 2b). The outputs of the context model and the hyperprior model \( h_a \) and \( h_s \) are used together for predicting the distribution parameters \( \mu \) and \( \sigma \).

While the above approaches focus on modeling the local dependencies, Qian et al. [37] propose a global reference model that captures long-range dependencies. This utilizes the most relevant previously decoded element for estimating distribution parameters of the current element. As shown in Fig. 2c, Qian et al. [37] use the global reference model \( r \) combined with the joint context and hyperprior model [32].

Motivation of Informer. Although utilizing the global dependencies is an innovative direction, the global reference model [37] does not fully exploit global information because only a single element among previously decoded ones is used. To improve the utilization of global information, we introduce a global hyperprior \( \hat{z}_g \) based on the attention mechanism [44] as shown in Figs. 2d and 3.

In addition, the global reference model [37] has an issue that the computational complexity increases quadratically with respect to the given image size. This is because in its self-attention-like mechanism, the most similar element to the current one is searched among all the previously decoded elements of the quantized latent representation. In order to avoid such an issue, in our global hyperprior modeling, we design an attention mechanism with a fixed number of query regardless of the image size. Learnable parameters of a fixed size called global tokens \( u \) are used as query.

3.3. Decomposition of hyperpriors

Extending the joint autoregressive and hierarchical priors [32], our entropy model, Informer, introduces two novel hyperpriors: a global hyperprior \( \hat{z}_g \) and a local hyperprior \( \hat{z}_l \). Fig. 3 shows a high-level overview of our hyperpriors in comparison with the existing hyperprior [8]. The hyperprior \( \hat{z} \) in the existing method reduces the spatial resolution of the latent representation \( y \) while retaining the number of channels. Due to the localized operations in CNNs, it uses limited local information containing only spatially adjacent elements.

We decompose the role of the hyperprior into two parts: local and global, as shown in Fig. 3. First, the global hyperprior \( \hat{z}_g \) consists of vectors having no spatial information. Note that \( \hat{z}_g \) is not limited to the local area, thus can handle the whole image area when modeling dependencies. In particular, the global dependencies are modeled in a content-dependent manner by using the attention mechanism [44]. Second, the local hyperprior \( \hat{z}_l \) models inter-channel dependencies in each spatial location to complement the lack of spatial components in \( \hat{z}_g \). It maintains the spatial resolution of the latent representation \( y \) while reducing the channel dimension. To extract and utilize the hyperpriors, we build Global Hyperprior Model (i.e., Global Hyper Encoder \( g_a \) and Global Hyper Decoder \( g_s \)) and Local Hyperprior Model (i.e., Local Hyper Encoder \( l_a \) and Local Hyper Decoder \( l_s \)) as shown in Fig. 2d.

3.4. Modeling of hyperpriors

Global hyperprior. Given the input latent representation \( y \in \mathbb{R}^{N \times H \times W \times C} \), where \( H, W, \) and \( C \) are the height, width, and the number of channels, respectively, the process of extracting the global hyperprior \( \hat{z}_g \) is illustrated in Global Hyper Encoder in Fig. 4. We define fixed-size global tokens \( u \in \mathbb{R}^{N \times C} \) as query of the multi-head attention layer of Global Hyper Encoder. \( N \) is a predefined parameter, which is fixed to eight in our final models. Note that the global tokens are learnable parameters that are determined through end-to-end training like other network parameters. With the multi-head attention block \( \text{MSA}(q, k, v) \), the MLP block \( \text{MLP}_1(\cdot) \), and the last MLP layer \( \text{MLP}_2(\cdot) \), the formulation of our Global Hyper Encoder is as follows:

\[
\begin{align*}
\hat{u}' &= \text{MSA}(u, y, y), \\
\hat{z}_g &= \text{MLP}_2(\text{MLP}_1(\hat{u}')),
\end{align*}
\]

where the normalization layers [3] are omitted for simplicity. \( \text{MLP}_1(\cdot) \) and \( \text{MLP}_2(\cdot) \) extract \( z_g \in \mathbb{R}^{N \times \hat{z}} \) for further modeling of inter-channel dependencies. After the
quantization operation $\lfloor \cdot \rfloor$, the global hyperprior $\hat{z}_g$ is obtained. Global Hyper Decoder, which is one linear layer, receives the global hyper prior $\hat{z}_g$ as the input, and generates $\psi_g \in \mathbb{R}^{N \times C}$.

**Local hyperprior.** In order to model inter-channel dependencies at each spatial location, we design Local Hyperprior Model by stacking $1 \times 1$ convolutional layers with the leaky ReLU activation [31], which is shown in the right part of Fig. 4. From the input $y$, Local Hyper Encoder extracts the local hyperprior $\hat{z}_l \in \mathbb{R}^{H \times W \times C}$, Local Hyper Decoder utilizes $\hat{z}_l$ and yields its output $\psi_l \in \mathbb{R}^{H \times W \times 2C}$.

### 3.5. Prediction of distribution parameters

As shown in Fig. 2d, we also employ Context Model $c$ that uses the previously decoded elements $\hat{y}_{<i}$. We adopt the same structure as in Minnen et al. [32], i.e., one $5 \times 5$ masked convolutional layer. For predicting the distribution parameters, i.e., mean $\mu \in \mathbb{R}^{H \times W \times 2C}$ and scale $\sigma \in \mathbb{R}^{H \times W \times 2C}$, we combine the outputs from Context Model, Local Hyperprior Model, and Global Hyperprior Model. Since the output of Global Hyperprior Model $\psi_g \in \mathbb{R}^{N \times 2C}$ does not have the spatial dimension, conventional feature combining methods such as concatenation or element-wise addition is inapplicable. Therefore, we propose a new Parameter Model with a multi-head attention-based combining method, which is shown in Fig. 5. The outputs of Context Model and Global Hyperprior Model ($\phi$ and $\psi_g$, respectively) are combined by using the multi-head attention block. In this block, $\phi$ is used as query, and $\psi_g$ is used as both key and value. After the multi-head attention block and following MLP block, the result is concatenated with the output of Local Hyperprior Model $\psi_l$ and the distribution parameters $\mu$ and $\sigma$ are generated via three $1 \times 1$ convolutional layers. The leaky ReLU activation functions are used after the first two $1 \times 1$ convolutional layers, which are omitted in Fig. 5 for simplicity. The formulation of Parameter Model is as follows:

$$\{\mu, \sigma\} = pm(\phi, \psi_g, \psi_l)$$

with $\phi = c(\hat{y}_{<i})$, $\psi_g = gh(y)$, and $\psi_l = lh(y)$, \hspace{1cm} (5)

where $pm(\cdot)$, $c(\cdot)$, $gh(\cdot)$, and $lh(\cdot)$ represent Parameter Model, Context Model, Global Hyperprior Model, and Local Hyperprior Model, respectively.

All the trainable models are learned using Eq. (1) with extending the rate term for the joint rate–distortion optimization to consider our hyperpriors $\hat{z}_l$ and $\hat{z}_g$:

$$\mathbb{E}_{x \sim p_x} \left[- \log_2 p_g(\hat{y}) - \log_2 p_{\hat{z}_l}(\hat{z}_l) - \log_2 p_{\hat{z}_g}(\hat{z}_g) \right],$$

\hspace{1cm} (6)

### 4. Experiments

All experiments are conducted on a PyTorch [36] based open-source library: CompressAI platform [9], which has recently been introduced for developing and evaluating learning-based image codecs.

**Training.** Our models are trained with various configurations of the Lagrange multiplier $\lambda$ and the distortion metric $d(\cdot, \cdot)$. We use MSE and $(1 - \text{MS-SSIM})$ for $d(\cdot, \cdot)$, and set $\lambda \in \{0.0018, 0.0035, 0.0067, 0.0130, 0.0250, 0.0483\}$ for MSE and $\lambda \in \{2.40, 4.58, 8.73, 16.64, 31.73, 60.50\}$ for MS-SSIM; thus, a total of 12 final models are obtained. We use the training images of the Triplet dataset, which is a subset of Vimeo-90K [49]. We use a batch size of 8 with 256×256 patches randomly cropped from the training images. All models are trained using the Adam optimizer [26] for 250 epochs. The learning rate starts at $10^{-4}$ and decreases to one-third at the 150th, 180th, 210th, and 240th epochs. For the MS-SSIM-optimized models, we finetune the MSE-optimized models with a learning rate of $10^{-5}$.

**Evaluation.** We evaluate our method on the Kodak [27] and Tecnick [2] datasets, which are commonly used for benchmarking learned image compression methods. The Kodak dataset consists of 24 images with a resolution of 768×512 pixels. The Tecnick dataset includes 100 images with a resolution of 1200×1200 pixels. At the encoding stage, we use the asymmetric numeral systems [19] for the entropy coding. To evaluate rate–distortion performance, we use the bits per pixel (bpp) and either the peak signal-to-noise ratio (PSNR) or MS-SSIM depending on the distortion metric $d(\cdot, \cdot)$. MS-SSIM is converted into decibels, i.e., $-10 \log_{10} (1 - \text{MS-SSIM})$ [8].

#### 4.1. Performance comparison

**Comparison with other entropy models.** To show the effectiveness of Informer, we compare its rate–distortion performance with that of the state-of-the-art image compression methods [28, 32, 37] in which the transformation
Table 1. Comparison of complexity of different entropy models at various image sizes. Each value means GFLOPs of each entropy model when the image having the corresponding resolution is used for entropy modeling. Note that we use the values in the original paper for global reference [37], since the official code is not released yet.

| Method                                | 320×240 | 480×360 | 640×480 | 768×512 | 1280×720 | 1920×1080 | 4096×2304 |
|---------------------------------------|---------|---------|---------|---------|----------|-----------|-----------|
| Context+Hyperprior [32]               | 2.73    | 6.14    | 10.91   | 13.96   | 32.73    | 73.64     | 335.13    |
| Context+Hyperprior+Global Reference [37] | 9.98    | 22.85   | 41.59   | 54.03   | 138.04   | 366.57    | 3296.99   |
| Informer (ours)                       | 2.69    | 6.04    | 10.73   | 13.73   | 32.15    | 72.34     | 329.17    |

Fig. 6 shows the relative rate savings compared to BPG [10] at different quality levels. Note that these graphs are generalized versions of the popular Bjøntegaard Delta (BD) chart [11] used for evaluation of image compression. As shown in Figs. 6a and b, our MSE-optimized models show significant rate savings over BPG, ranging from 11.84% to 19.27%. Compared to the baseline entropy model, i.e., “Context+Hyperprior (Minnen, 2018) [32]”, Informer significantly improves rate savings across all PSNR levels on both datasets. This shows that our global and local hyperpriors improve the accuracy of the learned entropy model. Even compared to the entropy model aiming at modeling global dependencies, i.e., “Context+Hyperprior+Global Reference (Qian, 2021) [37]”, Informer shows superiority across all PSNR levels on Kodak [27] with performance gaps from 1.74% up to 2.60%.

These results demonstrate the effectiveness of Informer’s capability to model joint global and local dependencies. Furthermore, when optimized with MS-SSIM (Fig. 6c), Informer outperforms all the other models on Kodak [27], which achieves rate savings by large margins up to 54.32%.

Complexity. We evaluate the performance of our Informer in terms of computational efficiency. For this, we measure the number of floating-point operations (FLOPs) required for various image sizes. Tab. 1 shows GFLOPs for our Informer and other entropy models [32, 37]. Informer shows the best performance across all image sizes. In particular, Informer does not suffer from the quadratic computational complexity problem that the “Context+Hyperprior+Global Reference [37]” model has. In other words, as the image resolution increases, GFLOPs of “Context+Hyperprior+Global Reference [37]” increases quadratically, being 10 times larger at the resolution of 4096×2304 than at the resolution of 1920×1080, while the complexity of Informer has a near-linear scale.

Qualitative performance. We compare visual quality of our decoded images with those of other traditional image codecs on the Kodak dataset [27]. For the image codecs, we employ JPEG [45] and BPG [10]. For a fair comparison, we encode the images under compression settings for as similar bpp values as possible. Fig. 7 shows that our method produces clear patterns (red boxes) and does not lose details (yellow boxes) compared to the other results.
Figure 7. Visual comparison of the decoded images by our methods and the other image codecs on “Kodim23” from Kodak [27].

Figure 8. Ablation studies of our method under different compression efficiency. (a) \(\lambda = 0.0067\) and (b) \(\lambda = 0.0483\).

4.2. Model analysis

In this section, we perform further analysis of our Informer. We utilize additional models trained using MSE for \(d(\cdot, \cdot)\) in two different bpp regions using \(\lambda = 0.0067\) and \(\lambda = 0.0483\), respectively.

Ablation study. To evaluate the contribution of the proposed hyperpriors in Informer, we conduct ablation studies. The rate–distortion performance with or without each hyperprior is shown in Fig. 8. It is shown that the original Informer shows the best results regardless of the bpp region. Introducing our global hyperprior leads to significant improvement compared to the entropy model using only the context prior (i.e., “Ours w/o Global & Local”), while our local hyperprior seems to be more helpful when combined with our global hyperprior than when used alone.

Analysis of decomposition. To validate the effectiveness of our approach to decomposing hyperpriors, we compare two different decomposition methods in Fig. 9: Informer (“Context+Local+Global”) and the existing hyperprior with our global hyperprior (“Context+Hyperprior+Global”). As a reference, we also provide the results of the existing context and hyperprior model [32] (“Context+Hyperprior”), which uses a single hyperprior without decomposition. We observe that Informer is better than the others for both bpp regions. The “Context+Hyperprior+Global” method shows even similar or slightly worse rate–distortion performance than the baseline (“Context+Hyperprior”) in the higher bpp region. We conjecture that this is due to the overlap in the roles of the existing hyperprior and our global hyperprior. The hyperprior-based approach requires additional bits to be encoded, and thus redundant roles can be an obstacle for improving performance due to excessive increase of bit usage, which implies that elaborate modeling of hyperpriors is important.

Analysis of combining methods. In Fig. 9, Informer using a variant of Parameter Model, denoted as “Context+Local+Global (L)”, is also evaluated, which combines the inputs to Parameter Model in a different way. More specifically, the local hyperprior \(\psi_l\) is used as query for the attention layer, and the context model \(\phi\) is concatenated after the MLP block. Our method shows better performance than this variant. In other words, it is more effective to combine the context prior with the global hyperprior when predicting the distribution parameters.

Analysis of attention. Fig. 10 visualizes example attention maps of our Global Hyper Encoder, which show where the corresponding global tokens pay attention to for capturing global dependencies. The results show that the global tokens successfully capture global information over the whole...
Figure 10. Visualization of the attention map used by the attention mechanism of the proposed Global Hyper Encoder. Two sample images from Kodak [27] are used, and two attention maps corresponding to two example global tokens are shown.

Figure 11. Model analysis with respect to the number of global tokens $N$, which is indicated by the number in parentheses. (a) $\lambda = 0.0067$ and (b) $\lambda = 0.0483$.

Image and different global tokens utilize different image regions in a content-dependent manner. In addition, we examine the effect of the number of global tokens on performance in Fig. 11. We observe that using four global tokens (“Ours (4)”) yields the best performance at the lower bpp region, while the best performance at the lower bpp region is obtained using eight global tokens (“Ours (8)”). This is probably due to the different degrees of remaining dependencies between the quantized latent representation $\hat{y}$. In other words, a large number of global tokens is effective to capture detailed dependencies for the higher bpp region.

4.3. Global hyperprior vs. global context prior

To examine whether the superior performance of Informer originates from the strong capability of the attention mechanism-based structure or introduction of the joint global and local hyperpriors, we evaluate another entropy model aiming at capturing global dependencies. Building on the context and hyperprior entropy model [32] as in Informer, we newly design Global Context Model using the masked multi-head self-attention mechanism [44]. As shown in Fig. 12a, Global Context Model receives the output of Context Model $\phi$ and updates it by attending the previously decoded elements. Through the process of the attention mechanism, Global Context Model considers multiple references, and thus it can be seen as a generalized version of the global reference model [37] utilizing only the most relevant reference. Tab. 2 shows that Informer is superior to the method using Global Context Model. We argue that introducing the self-attention mechanism is not a silver bullet for image compression, but careful design consideration should be accompanied with it.

| $\lambda$ | Method             | Rate (bpp) | PSNR (dB) |
|---------|--------------------|------------|-----------|
| 0.0067  | Informer           | 0.2763     | 31.3254   |
|         | Global context     | 0.2842     | 31.2947   |
| 0.0483  | Informer           | 0.8514     | 36.9739   |
|         | Global context     | 0.8718     | 36.9134   |

Table 2. Comparison of the different methods of introducing global vision to learned image compression. The “Global context” model introduces global vision to Context Model using the masked self-attention mechanism. The results are obtained at 160 epochs.

5. Conclusion

We proposed a novel entropy model (Informer) for learned image compression. Based on the previous joint autoregressive and hierarchical priors [32], we proposed two different hyperpriors for modeling remaining dependencies in the quantized latent representation, one for global dependencies, and the other for local dependencies. We showed that Informer outperforms existing entropy models in terms of rate–distortion performance with computational efficiency. Beyond the existing CNN-based localized dependency modeling methods, Informer presents a completely new approach that effectively utilizes both global and local information in a content-dependent manner using the attention mechanism.

6. Limitation

Informer cannot be parallelized in its decoding process, which is due to the inherent limitation of the autoregressive prior that utilizes only previously decoded elements. We expect that this problem can be addressed by combining our approach with the channel-wise autoregressive model [33] or bidirectional context model [22].
References

[1] Eirikur Agustsson and Lucas Theis. Universally quantized neural compression. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), 2020.

[2] Nicola Asuni and Andrea Giachetti. TESTIMAGES: a large-scale archive for testing visual devices and basic image processing algorithms. In Proceedings of the Smart Tools and Apps for Graphics (STAG), 2014.

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. arXiv preprint arXiv:1607.06450, 2016.

[4] Song Bai, Philip Torr, et al. Visual Parser: Representing part-whole hierarchies with transformers. arXiv preprint arXiv:2107.05790, 2021.

[5] Johannes Ballé, Philip A Chou, David Minnen, Saurabh Singh, Nirad Singh, Eirikur Agustsson, Sung Jin Hwang, and George Toderici. Nonlinear transform coding. IEEE Journal of Selected Topics in Signal Processing, 15(2):339–353, 2020.

[6] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. Density modeling of images using a generalized normalization transformation. In Proceedings of the International Conference on Learning Representations (ICLR), 2016.

[7] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. End-to-end optimized image compression. In Proceedings of the International Conference on Learning Representations (ICLR), 2017.

[8] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. In Proceedings of the International Conference on Learning Representations (ICLR), 2018.

[9] Jean Béган, 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.

[10] Fabrice Bellard. BPG image format. http://bellard.org/bpg/.

[11] Gisle Bjøntegaard. Calculation of average PSNR differences between RD-curves. VCEG-M33; 2001.

[12] 2021 Worldwide Image Capture Forecast: 2020 – 2025. https://riseaboveresearch.com/rar-reports/2021-worldwide-image-capture-forecast-2020-2025/.

[13] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In Proceedings of the European Conference on Computer Vision (ECCV), 2020.

[14] Chun-Fu Chen, Quanfu Fan, and Rameswar Panda. CrossViT: Cross-attention multi-scale vision transformer for image classification. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2021.

[15] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. Deep residual learning for image compression. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPRW), 2019.

[16] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. Learned image compression with discretized gaussian mixture likelihoods and attention modules. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[17] Ze Cui, Jing Wang, Shangyin Gao, Tiansheng Guo, Yihui Feng, and Bo Bai. Asymmetric gained deep image compression with continuous rate adaptation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2021.

[18] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In Proceedings of the International Conference on Learning Representations (ICLR), 2020.

[19] Jarek Duda. Asymmetric numeral systems: entropy coding combining speed of Huffman coding with compression rate arithmetic coding. arXiv preprint arXiv:1311.2540, 2013.

[20] Ge Gao, Pei You, Rong Pan, Shunyuan Han, Yuanyuan Zhang, Yuchao Dai, and Hojae Lee. Neural image compression via attentional multi-scale back projection and frequency decomposition. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2021.

[21] Vivek Goyal. Theoretical foundations of transform coding. IEEE Signal Processing Magazine, 18(5):9–21, 2001.

[22] Dailan He, Yaoyan Zheng, Baocheng Sun, Yan Wang, and Hongwei Qin. Checkerboard context model for efficient learned image compression. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2021.

[23] Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2021.

[24] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. Neural Networks, 2(5):359–366, 1989.

[25] Nick Johnston, Damien Vincent, David Minnen, Michele Covell, Saurabh Singh, Troy Chinen, Sung Jin Hwang, Joel Shor, and George Toderici. Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[26] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[27] The Kodak PhotoCD dataset. http://r0k.us/graphics/kodak/.

[28] Jooyoung Lee, Seunghyun Cho, and Seung-Kwon Beach. Context-adaptive entropy model for end-to-end optimized image compression. In Proceedings of the International Conference on Learning Representations (ICLR), 2019.
Supplementary Material

A. Effectiveness of Informer

In our main paper, we evaluated our entropy model (Informer) with the encoder-decoder structure in Minnen et al. [32]. Additionally, in this supplementary material, we evaluate Informer with another encoder-decoder structure in Cheng et al. [16], which consists of residual blocks and attention modules. For the baseline entropy model [32], we use the result values reported in the CompressAI Github page\(^1\).

As shown in Fig. A.1, Informer shows superiority across all PSNR levels on Kodak [27] with performance gaps from 3.03% up to 5.25%. From this result, we observe that our proposed Informer effectively models remaining dependencies in the quantized latent representation regardless of the encoder-decoder structure.

![Figure A.1](https://github.com/InterDigitalInc/CompressAI/blob/master/results/kodak/compressai-cheng2020-attn_mse_cuda.json)

Figure A.1. Performance of different image compression methods. Each curve means the rate savings (%) relative to BPG [10] at different PSNR levels. Larger values mean better performance. The results of the MSE-optimized methods are averaged over Kodak [27].

\(^1\)https://github.com/InterDigitalInc/CompressAI/blob/master/results/kodak/compressai-cheng2020-attn_mse_cuda.json
B. More implementation details

We illustrate the overall diagram of the learned image compression method with Informer in Fig. B.1. In addition, architectural details of Informer are summarized in Tab. B.1.

Figure B.1. Overall diagram of the learned image compression method with the proposed entropy model, Informer. The white blocks are data tensors, the blue blocks are learned models, the red blocks are entropy coding, and the gray circles are quantization operations. Informer can be combined with any transformation part (consisting of an encoder \( f_a \) and a decoder \( f_s \)). Informer jointly optimizes Context Model (capturing dependencies in the previously decoded local context \( \hat{y}_{<i} \)), Local Hyperprior Model (capturing inter-channel dependencies at each spatial location), Global Hyperprior Model (capturing global dependencies across the whole image region), and Parameter Model (predicting the distribution parameters \( \mu \) and \( \sigma \)). By utilizing the predicted \( \mu \) and \( \sigma \) by Informer, the quantized latent representation \( \hat{y} \) is encoded into a bitstream using an entropy encoder (EC) and the bitstream is decoded by an entropy decoder (ED). In addition to \( \hat{y} \), the proposed local hyperprior \( \hat{z}_l \) and global hyperprior \( \hat{z}_g \) are also encoded using entropy coding. For this, we employ the factorized entropy model [8], where all elements of \( \hat{z}_l \) with the same channel index are assumed to follow the same distribution and all elements of \( \hat{z}_g \) are assumed to follow different distributions.

| Context Model | Local Hyper Encoder | Local Hyper Decoder | Global Hyper Encoder | Global Hyper Decoder | Parameter Model (1) | Parameter Model (2) |
|---------------|---------------------|---------------------|----------------------|----------------------|---------------------|---------------------|
| Masked: 5×5 c384 s1 | Conv: 1×1 c48 s1 | Conv: 1×1 c98 s1 | MHA: c192 h4 | Conv: 1×1 c384 s1 | MHA: c384 h8 | Conv: 1×1 c640 s1 |
|                | Leaky ReLU          | Leaky ReLU          | Linear: c768        | Linear: c192        | Linear: c51536      | Leaky ReLU          |
|                | Conv: 1×1 c12 s1    | Conv: 1×1 c384 s1  | GELU                 | Conv: 1×1 c384 s1   | Linear: c384        | Conv: 1×1 c512 s1   |

Table B.1. Architectural details of Informer when \( C = 192 \) and \( N = 8 \). Convolutional layers are represented by the “Conv” prefix followed by the kernel size, number of channels and stride (e.g., the first layer of the local hyper encoder uses 1×1 kernels with 48 channels and a stride of one). The “Masked” prefix means the masked convolutional layer as in [35]. The “MHA” prefix means the multi-head attention blocks as in [18], followed by the number of channels and the number of heads. The “Linear” prefix means the linear layers followed by the number of channels.
C. Rate–distortion curves

Fig. C.1 shows rate–distortion curves corresponding to Fig. 6 in the main paper. For traditional methods, we use JPEG [45], JPEG2000 [38], and BPG [10]. For learned methods, we use those of Minnen et al. [32], Lee et al. [28], and Qian et al. [37].

Figure C.1. Rate–distortion curves. Top: MSE-optimized models on Kodak [27]. Middle: MSE-optimized models on Tecnick [2]. Bottom: MS-SSIM-optimized models on Kodak [27].
D. More qualitative results

Figs. D.1 to D.5 show qualitative performance comparison between our method and other traditional image codecs. As in the main paper, for fair comparisons, we encode the images under compression settings for as similar bpp values as possible. Overall, our methods exhibit better qualitative performance by restoring images more clearly, not reconstructing wrong content that does not exist in the original image, and producing patterns more similar to those of the original image.

Figure D.1. Visual comparison of the decoded images by our methods and the other image codecs on “Kodim03” from Kodak [27].

Figure D.2. Visual comparison of the decoded images by our methods and the other image codecs on “Kodim15” from Kodak [27].
Figure D.3. Visual comparison of the decoded images by our methods and the other image codecs on “Kodim04” from Kodak [27].

Figure D.4. Visual comparison of the decoded images by our methods and the other image codecs on “Kodim09” from Kodak [27].

Figure D.5. Visual comparison of the decoded images by our methods and the other image codecs on “Kodim17” from Kodak [27].