Another Way to the Top: Exploit Contextual Clustering in Learned Image Coding

Yichi Zhang[1,2], Zhihao Duan[2], Ming Lu[3], Dandan Ding[1]*, Fengqing Zhu[2], Zhan Ma[3]

[1]Hangzhou Normal University, Hangzhou, Zhejiang, China
[2]Purdue University, West Lafayette, Indiana, U.S.
[3] Nanjing University, Nanjing, Jiangsu, China
yichizhang@ieee.org, duan90@purdue.edu, minglu@nju.edu.cn, DandanDing@hznu.edu.cn, zhu0@purdue.edu, mazhan@nju.edu.cn

Abstract

While convolution and self-attention are extensively used in learned image compression (LIC) for transform coding, this paper proposes an alternative called Contextual Clustering based LIC (CLIC) which primarily relies on clustering operations and local attention for correlation characterization and compact representation of an image. As seen, CLIC expands the receptive field into the entire image for intra-cluster feature aggregation. Afterward, features are reordered to their original spatial positions to pass through the local attention units for inter-cluster embedding. Additionally, we introduce the Guided Post-Quantization Filtering (GuidedPQF) into CLIC, effectively mitigating the propagation and accumulation of quantization errors at the initial decoding stage. Extensive experiments demonstrate the superior performance of CLIC over state-of-the-art works: when optimized using MSE, it outperforms VVC by about 10% BD-Rate in three widely-used benchmark datasets; when optimized using MS-SSIM, it saves more than 50% BD-Rate over VVC. Our CLIC offers a new way to generate compact representations for image compression, which also provides a novel direction along the line of LIC development.

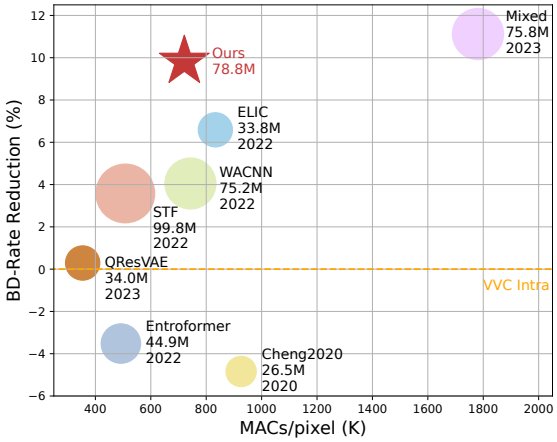


Figure 1: Coding Performance versus End to End Complexity across a variety of state-of-the-art LIC methods. MACs/pixel is calculated in the end-to-end manner, and BD-Rate is averaged in the Kodak dataset.

Introduction

Lossy image compression is one of the most fundamental issues in information theory and signal processing. Most existing methods for lossy image compression follow the scheme of *transform coding* (Goyal 2001), where images are transformed to a latent space for de-correlation and energy compression, followed by quantization and entropy coding. Historically, traditional codecs such as JPEG (Wallace 1992), BPG (Sullivan et al. 2012), and VVC (Bross et al. 2021) have utilized simple linear transforms (e.g., the discrete cosine transform) to accomplish this goal.

In recent years, learned image compression (LIC) methods have achieved superior performance over the traditional codecs (Koyuncu et al. 2022; Liu, Sun, and Katto 2023). Central to the success of LIC is the utilization of non-linear transforms such as convolutional neural networks (CNNs) and Transformer modules. Early works stack standard convolutional layers for feature extraction and image reconstruction (Ballé, Laparra, and Simoncelli 2017; Ballé et al. 2018). Later, deformable convolutions (Zhu et al. 2019), octave convolutions (Chen, Xu, and Wang 2022), and asymmetric convolutions (Tang et al. 2023) are developed to improve the standard convolutions. Recent works also introduce Transformers into CNNs (Liu, Sun, and Katto 2023). Although these convolution and attention-based methods consistently improve the rate-distortion (RD) performance of LIC, they often come at the expense of increased computational complexity, as depicted in Figure 1.

In contrast, this paper aims at improving the transform and quantization in the (non-linear) transform coding framework without increasing the overall computational complexity. We first propose "Contextual Clustering based LIC (CLIC)", an alternative approach to convolution and Transformer-based methods, to exploit and characterize the spatial correlation within an image from a new perspective. Specifically, inspired by (Ma et al. 2023b), we break the regular convolution operations on pixels, but reorganize all pixels in an image into several categories according to their similarities and apply Multilayer Perceptron (MLP) for intra-cluster correlation characterization. Furthermore, local attention units like spatial attention (Woo et al. 2018) and

*Corresponding author

channel attention (Hu, Shen, and Sun 2018) are embedded for inter-cluster exploration to augment the coding performance. In this way, the CLIC approach greatly reduces convolutions that require higher parameters and MACs by replacing them with simple linear and attention operations.

We also notice that the quantization operation following the transform step inevitably introduces quantization errors, adversely impacting the quality of reconstructions. To this end, filters are usually appended to decoded images for quality improvement (Li et al. 2019; Zhang et al. 2024). This, however, ignores the implicit propagation and accumulation of quantization error in the decoding process, which may limit the filtering performance (Fu et al. 2023a). To tackle this issue, we develop a Guided Post-Quantization Filtering method that compensates for the error at the beginning of decoding to prevent error propagation. It utilizes a set of coefficients supervised by the true errors to guide the decoder for filtering, enabling content-adaptive processing. By encoding these coefficients into the bitstream, the rate-distortion performance is improved with negligible complexity overhead.

Extensive experiments demonstrate the superior performance of our CLIC, as illustrated in Figure 1. Replacing the clustering operations with conventional convolutions in CLIC not only results in a slight decrease in coding performance but also increases computational complexity by 25%. Overall, CLIC provides an alternative and promising solution for learned image compression beyond merely incremental performance improvement.

Our contributions are summarized as follows:

- We propose the Contextual Clustering based approach, termed CLIC, mostly utilizing linear-based clustering and local attention to characterize pixel correlations in an image-level receptive field, outperforming convolution-based and window attention-based networks;
- We propose Guided Post-Quantization Filtering (GuidedPQF) to mitigate the propagation and accumulation of quantization errors while enabling content-adaptive processing at the initial decoding stage, improving rate-distortion performance;
- Our CLIC achieves around 10% BD-Rate reduction over VVC on various image test sets, surpassing the performance of most existing approaches. Extensive experiments are conducted to verify our findings and demonstrate the effectiveness of CLIC.

Related Work

To achieve RD optimization, two crucial techniques, transform coding and context modeling, are used for the generation and the entropy coding of image latent features in LIC, respectively. We will next review these two techniques.

Transform Coding

Recently, learning-based transforms (Chadha and Andreopoulos 2021; Ma et al. 2023a; Zhang, Deng, and Li 2022) have shone in both rules-based and learned-based image compression. Regarding the transforms in LIC, stacked convolution layers are widely used to transform an image into another latent space for sparse representation. For example, Ballé *et al.* (Ballé, Laparra, and Simoncelli 2017) applied stacked convolutions in the VAE, together with Generalized Divisive Normalization (GDN), Inverse GDN (IGDN), for the transform function. In addition to convolutions, attention modules (Cheng et al. 2020) were introduced for compact representation of images, offering the first LIC method that obtained comparable performance to VVC. Then, convolutions and non-local attention modules were jointly utilized to derive latent features and hyperpriors and achieved impressive results (Chen et al. 2021).

Moreover, the prevalence of self-attention-based Transformers has inspired many researchers to investigate the use of Transformers for nonlinear transforms. Lu *et al.* (Lu et al. 2022b) introduced neural transformation units, which combined a Swin Transformer Block and a convolutional layer for the compact representation of images. Zhu *et al.* (Zhu, Yang, and Cohen 2022) employed Swin Transformer while Zou *et al.* (Zou, Song, and Zhang 2022) utilized a symmetrical Transformer for LIC. More recently, Liu *et al.* (Liu, Sun, and Katto 2023) mixed the structures of convolutions and Swin Transformer, achieving state-of-the-art results.

Context Modeling

Context Model significantly affects coding efficiency. In rules-based codecs, context-adaptive variable-length coding (CAVLC) (Moon, Kim, and Kim 2005) and context-adaptive binary arithmetic coding (CABAC) (Sze and Budagavi 2012) are widely used to reduce redundancy by exploring the statistical correlation across coding symbols. In LIC, context models are also devised with the same goal of coding symbols using the lowest bitrate. Typical methods include the autoregressive context model (Minnen, Ballé, and Toderici 2018) and its variants (Qian et al. 2021; Koyuncu et al. 2022; Kim, Heo, and Lee 2022; Qian et al. 2022).

However, these methods require long decoding times due to the nature of serial processing, hindering their use in practice. To this end, parallel context modeling methods, which are more friendly to real applications, are developed. The checkerboard model (He et al. 2021) is a typical tool, in which the anchor content is encoded independently while the non-anchor content is encoded at a lower cost depending on the anchor content priors. Later, a generalized checkerboard (Lu et al. 2022a) and a dual spatial prior model (Guo-Hua et al. 2023) are introduced.

In addition to exploiting spatial correlation in an image, numerous works have been devoted to exploring the correlation across channels. The channel conditional model was devised to divide channels into slices for parallel processing (Minnen and Singh 2020) and later it was combined with the autoregressive and hierarchical prior entropy model to form a cross-channel context model (Ma et al. 2021). Recently, He *et al.* (He et al. 2022) noticed the uneven distribution of information among channels and proposed a channel-wise model with uneven grouping, termed the space-channel context model (SCCTX). Due to the well-balanced coding efficiency and time complexity of SCCTX, we directly use SCCTX for context modeling in this work.

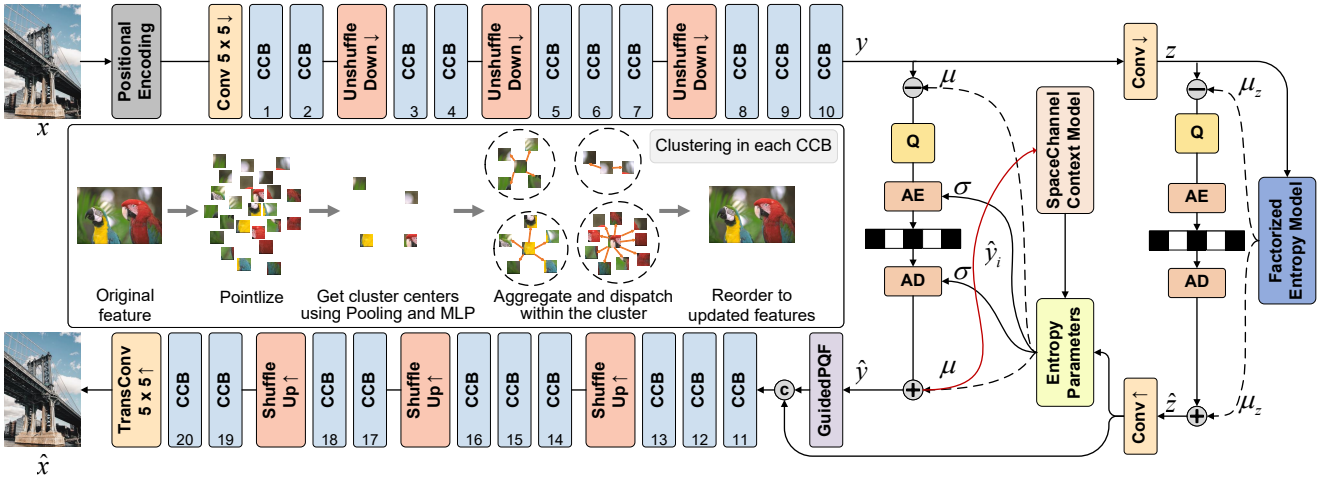


Figure 2: The overall architecture of the proposed method. ↓/↑ indicates downsampling/upsampling operations. The hyper encoder and decoder each consist of five convolutional layers, succeeded by a GELU function, except for the final layer. The channel of z is 192. Three hyper decoders are utilized for the mean, scale, and latent feature, respectively.

Remarks. Previous studies have extensively demonstrated the outstanding capability of convolutions for the transform coding of LIC. However, the mountaintop embraces diverse paths that lead to it. While convolutions offer promising coding efficiency, we present another elegant way by leveraging contextual clustering in LIC, which achieves even higher performance than previous works.

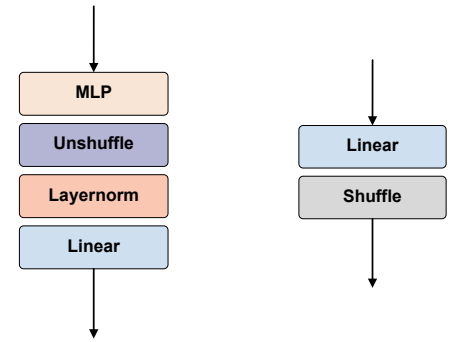
Proposed Method

Overview

An overview of the proposed CLIC is illustrated in Figure 2. Both the main analysis transform module and the main synthesis transform module comprise four stages, each containing one downsampling/upsampling operation and a set of Contextual Clustering Blocks (CCBs). The downsampling operation stacks MLP, Unshuffle, Layernorm, and Linear layers, while the upsampling operation only goes through a linear layer and a shuffle layer, as shown in Figure 3. After downsampling/upsampling are stacked CCBs to exploit spatial correlations across pixels for a compact representation of the input image.

At the beginning of the analysis transform, we cascade the attribute features (RGB color) of each pixel with its position features (Cartesian coordinate system (X, Y) coordinates). In this way, an image $I \in \mathbb{R}^{H\times W\times 3}$ is transformed into $P \in \mathbb{R}^{n\times 5}$ points, $n = H \times W$, with each point consisting of its RGB attribute (r, g, b) and position information (x, y) (for Positional Encoding). For the context modeling, we directly combine the checkboard spatial model and uneven channel-wise to generate SCCTX, and the latent feature for reconstruction is also utilized to fully explore the side information. More details can refer to (He et al. 2022; Hu, Yang, and Liu 2020).

At the beginning of the main synthesis transform, GuidedPQF is applied to compensate for the quantization errors



(a) Downsampling (b) Upsampling

Figure 3: Strucuture of Down / Up sampling

in the latent feature domain. Afterward, features are progressively upsampled and decoded. In the end, the reconstructed image is produced through a 5×5 transposed convolution.

In the next, we will detail the algorithms and structures of CCB and GuidedPQF.

Contextual Clustering Block

As depicted in Figure 4, CCB is built on the structure of Metaformer (Yu et al. 2022), which consists of two layernorm (Ba, Kiros, and Hinton 2016), Token Mixer, Channel Mixer, and a residual connection. In our CLIC, the Token Mixer is implemented using the clustering approach (Achanta et al. 2012; Ma et al. 2023b), followed by the spatial attention unit for feature enhancement. The Channel Mixer is realized using a simple MLP, followed by the channel attention unit. As observed and demonstrated in our experiments, clustering and attention units play crucial roles within each CCB.

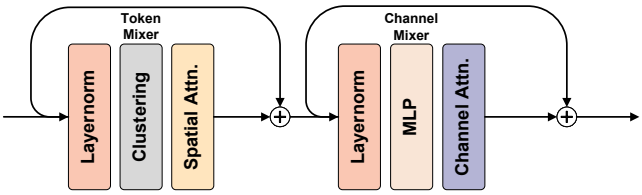


Figure 4: The structure of CCB. The MLP contains two linear layers and a GELU activation function.

Contextual Clustering. In contrast to convolutions that treat the image as a regular grid and operate in a fixed-size receptive field for processing, CCB considers the image as an unordered set of points (each point P_i is a vector) and conducts feature extraction through clustering. Each point contains its attribute features and position features.

The clustering results greatly affect the performance of CCB. After extensive experiments, we resort to cosine similarity for clustering. The corresponding clustering algorithm is described as follows:

1. First, add the global context to all points by global average pooling:

$$P_i = P_i + \gamma \cdot \frac{1}{n} \sum_{i=1}^{n} P_i, \tag{1}$$

where γ is a learnable parameter to scale the global context dynamically. Then, all the pixel points are linearly transformed.

2. Get c clustering centers by average pooling the feature map (i.e., all points P). In our experiments, we choose $c = 2 \times 2 = 4$ by default (we discuss the choice of c in our ablation study). Then, a 2-layer MLP is used to predict the offset of each clustering center.
3. Calculate the cosine similarity matrix for all points with c clustering centers, and the points most similar to each center are grouped.
4. Within each cluster, points are aggregated based on their similarity to the clustering center. Suppose we have m points for a cluster. We first linearly transform all the points to get their value vectors $v_i, \forall i \in \{1, 2, ..., m\}$, and their center point c_v is obtained in the same way as in step 2. The output feature F for this cluster is computed by:

$$F = \frac{1}{1+m} \left(c_v + \sum_{i=1}^{m} \text{sigmoid}\left(\alpha \cdot s_i + \beta\right) \cdot v_i \right), \tag{2}$$

where α and β are learnable parameters, and s_i is the cosine similarity between the i-th point and the center. According to (Ma et al. 2023b), this gives better performance in a stable manner due to the non-negativity of its similarity. c_v is used to emphasize the attribution of its class. The normalization term $\frac{1}{1+m}$ controls the output feature magnitude, where we use $(1 + m)$ as the denominator to avoid division by zero when there is no point in a cluster (i.e., $m = 0$).

Figure 5: Clustering results on Kodak. The same color mask represents the same category. Clustering results are obtained from the final CCB.

5. The aggregated features are then adaptively assigned to each point in the cluster. For each point P_i, it is updated by $P_i \leftarrow P_i + \text{Linear}(s_i \cdot F)$. A linear layer is applied to match the dimension of F and P_i.

Moreover, we apply slightly different processing on odd-numbered and even-numbered CCBs. The odd-numbered CCBs exactly follow the above steps. For the even-numbered CCB, we divide features into two groups for separate processing in a checkboard manner. When one group passes through CCB, the other group is masked as zero. Such a checkerboard pattern enables the separation of features that have already been clustered together under one category in the odd-numbered CCBs, so as to generate new clustering results in the even-numbered CCBs at the same scale. As such, CLIC is able to derive various clustering features and more effectively utilize contextual information to improve coding efficiency.

Reordering. As seen, the clustering operation typically exploits point relationships within each cluster. Furthermore, after clustering, latent features obtained are reordered to the image shape before clustering according to their positions. Attention units are then applied for further processing.

Attention Enhancement. The contextual clustering approach primarily focuses on intra-cluster interactions, which limits its local neighborhood correlations. To this end, we further augment the attention mechanism after contextual clustering to exploit the inter-cluster correlations. As shown in Figure 6, two local attention units, including Spatial Attention (SA) and Channel Attention (CA), are applied in each CCB following Token Mixer and Channel Mixer, respectively. In this way, both intra-cluster and inter-cluster correlations are included in CCB for efficient representation.

Remarks. CCB systematically rearranges all points in an image, enabling global correlation perception and aggregation in a predefined clustering manner, as the clustering results visualized in Figure 5. Unlike conventional Transformers that rely on computationally intensive matrix-based operations, the clustering algorithm only requires linear operations, potentially offering practical advantages on spe-

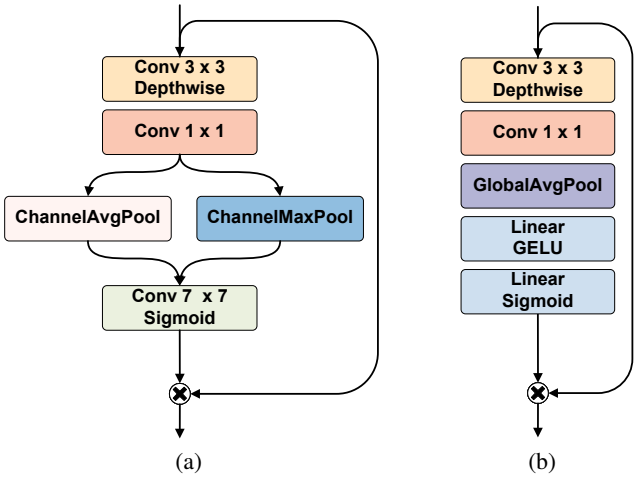


Figure 6: Modular components: (a) Spatial Attention; (b) Channel Attention.

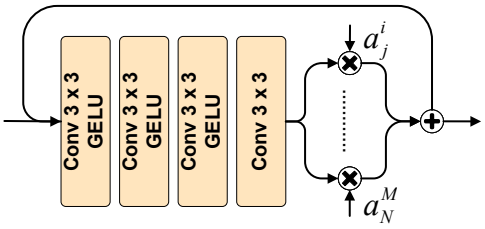


Figure 7: The structure of GuidedPQF.

cific platforms. The subsequent local attention units further strengthen the valuable features across clusters. As a result, CCB and local attention units collaboratively contribute to the compact image representation.

Guided Post-Quantization Filtering

All features denoted as y in Figure 2 undergo quantization for entropy coding, which inevitably introduces quantization errors. Such errors will be propagated and accumulated from scale to scale during the decoding process, severely impairing the reconstructed quality (Fu et al. 2023a). To address this issue, we propose preventing such error propagation from the start, i.e., a GuidedPQF is applied on the dequantized $\hat{y}$ to compensate for the quantization errors.

To this end, the goal of GuidedPQF is to minimize the distance between the estimated and true quantization errors:

$$e = \|\tilde{\epsilon} - \epsilon\|^2 = \|F(\hat{y};\theta) - \epsilon\|^2, \tag{3}$$

where $\tilde{\epsilon} = \tilde{y} - \hat{y} = F(\hat{y};\theta)$ represents the the estimated quantization error output from GuidedPQF $F(\cdot)$. $\epsilon = y - \hat{y}$ denotes the true quantization error.

In contrast to previous studies that directly employ a neural model to realize the filter $F(\cdot)$, GuidedPQF utilizes a set of coefficients from the encoder to guide the filter for content-adaptive processing. As illustrated in Figure 7, given the dequantized features $\hat{y} = \{\hat{y}_i\}_{i=1}^M$, GuidedPQF will generate N filtering candidates $\mathbf{C}_1^i, \mathbf{C}_2^i, \cdots, \mathbf{C}_N^i$ for each $\hat{y}_i$ and weigh them to derive $\tilde{\epsilon} = \{\tilde{\epsilon}_i\}_{i=1}^M$ and then $\tilde{y} = \{\tilde{y}_i\}_{i=1}^M$:

$$\tilde{y}_i = \tilde{\epsilon}_i + \hat{y}_i = a_1^i \mathbf{C}_1^i + a_2^i \mathbf{C}_2^i + \cdots + a_N^i \mathbf{C}_N^i + \hat{y}_i, \tag{4}$$

where a_j^i, $j \in \{1, 2, ..., N\}$ are weighting coefficients that can be signaled in bitstream.

Based on Eq. (3) and Eq. (4), given $\epsilon = \{\epsilon_i\}_{i=1}^M$, a_j^i can be obtained by least square optimization (Ding et al. 2023):

$$\left[a_1^i, a_2^i, \cdots, a_N^i\right]^{\mathrm{T}} = \left(\mathbf{C}^{i\mathrm{T}}\mathbf{C}^i\right)^{-1}\mathbf{C}^{i\mathrm{T}}\epsilon_i, \tag{5}$$

where $\mathbf{C}^i = [\mathbf{C}_1^i, \mathbf{C}_2^i, \cdots, \mathbf{C}_N^i]$ stacks vectorized $\mathbf{C}_j^i$s.

Substituting Eq. (5) into Eq. (3), we can derive the reconstruction error e_i:

$$e_i = \|\hat{\epsilon}_i - \epsilon_i\|^2 = |\epsilon_i|^2 - \epsilon_i^{\mathrm{T}}\mathbf{C}^i\left(\mathbf{C}^{i\mathrm{T}}\mathbf{C}^i\right)^{-1}\mathbf{C}^{i\mathrm{T}}\epsilon_i. \tag{6}$$

As a result, for $\hat{y}$ with M channels, the objective function of our GuidedPQF can be described as:

$$\mathcal{L}_{\mathcal{PQF}} = \sum_{i=1}^{M}\left\{\cancel{|\epsilon_i|^2} - \epsilon_i^{\mathrm{T}}\mathbf{C}^i\left(\mathbf{C}^{i\mathrm{T}}\mathbf{C}^i\right)^{-1}\mathbf{C}^{i\mathrm{T}}\epsilon_i\right\}. \tag{7}$$

Since $|\epsilon_i|^2$ is a constant for each $\hat{y}_i$, we can directly remove it. Finally, the overall loss function is a summation of $\mathcal{L}_{\mathcal{PQF}}$ and the common rate-distortion loss function:

$$\mathcal{L} = \mathcal{R} + \lambda \cdot \mathcal{D} + \lambda_1 \cdot \mathcal{L}_{\mathcal{PQF}}, \tag{8}$$

where the rate $\mathcal{R}$ and distortion $\mathcal{D}$ are computed following the standard practice in Hyperprior models (Ballé et al. 2018). λ_1 is weighting factors adjusting magnitude orders and is set to 1 in our method. As the weighting coefficients a_j^is are obtained from the true error ϵ, they have to be passed to the decoder by consuming certain bitrates. We use fix-length coding (four-bits representation) to encode a_j^is and set $N = 2$ to limit their bitrate increase.

Experimental Results

Experimental Settings

Training. We use Flicker2W (Liu et al. 2020b) and LIU4K (Liu et al. 2020a) as our training sets. We first scaled the longer side of the images in LIU4K to 2000 pixels with the same aspect ratio, and then randomly cropped 256×256 patches for training. Following Liu *et al.* (Liu et al. 2020b), 99% of the images were used for training, and the remaining 1% were used for validation. Following the settings of CompressAI (Bégaint et al. 2020), we set $\lambda \in \{18, 35, 67, 130, 250, 483\} \times 10^{-4}$ for MSE optimized model and $\lambda \in \{2.40, 4.58, 8.73, 16.64, 31.73, 60.50\}$ for MS-SSIM optimized model. We trained each model with Adam optimizer (Kingma and Ba 2014) with $\beta_1 = 0.9$, $\beta_2 = 0.999$. Each model was trained for 300 epochs with a batch size of 8 and an initial learning rate of $1e^{-4}$. We used the ReduceLRonPlateau lr scheduler with a patience of 5 and a factor of 0.5. The first 150 epochs are called Stage 1 and the next 150 epochs are called Stage 2. When switching from Stage 1 to Stage 2, the learning rate is newly set to $1e^{-4}$. We use

mixed quantizers to train the channel-conditional models. We use universal quantization (U-Q) (Choi, El-Khamy, and Lee 2019) to generate U-Q(y) for the context model and use differentiable soft quantization (DS-Q) (Gong et al. 2019) to generate DS-Q(y) as input to the decoder. Also, following previous work (Minnen, Ballé, and Toderici 2018; Minnen and Singh 2020; He et al. 2022) and community discussions[1], we do not encode $\lceil y \rfloor$ as a bitstream, instead, we encode each $\lceil y - \mu \rfloor$. When reverting the encoded symbols, we revert them to $\lceil y - \mu \rfloor + \mu$, which enables the single Gaussian entropy model to yield better results.

All training was performed on a computer with an RTX4090 GPU, i9-13900K CPU, and 64G RAM. Ablation experiments were performed on a computer with an RTX3090 GPU, i7-9700K CPU, and 64G RAM.

Testing. Three widely-used benchmark datasets, including Kodak[2], Tecnick[3], and CLIC 2022[4], are used to evaluate the performance of the proposed method.

Quantitative Results

We compare our proposed method with prevalent learning-based image compression models including Cheng2020 (Cheng et al. 2020), Entroformer (Qian et al. 2022), NeuralSyntax (Wang et al. 2022), QResVAE (Duan et al. 2023), GLLMM (Fu et al. 2023b), STF (Zou, Song, and Zhang 2022), WACNN (Zou, Song, and Zhang 2022), ELIC (He et al. 2022), Contextformer (Koyuncu et al. 2022), and Mixed (Liu, Sun, and Katto 2023) and rules-based VVC (Bross et al. 2021). We use VVC reference software VTM-18.0 under All Intra configuration as the anchor to calculate BD-Rate. RD points of ELIC, Contextformer, and Mixed are digitized from the figures in their original papers or websites since they are not open-source.

Table 1 reports the BD-Rate reduction of each method against the VVC anchor on three datasets. Our proposed method achieves near-best performance on each dataset, comparable to Mixed, on average 9.83% on Kodak, 11.00% on Tecnick, and 10.05% on CLIC 2022. Figure 8 further plots RD curves of all methods. Moreover, our method consistently maintains around 10% BD-Rate gain over VVC on the other two datasets, showcasing its strong generalization on various content and resolutions.

Our CLIC gains slightly lower than Mixed when optimized with MSE. This occurs mainly because: 1) our training dataset has only 21,600 images while Mixed's has 300,000 images; 2) Mixed stacks massive Convolutions and Transformers at the expense of highly intensive complexity, which is 2.47× of our MACs.

In addition, the coding performance of MS-SSIM optimized models is presented in Table 1. Our method achieves the best results and is the only one that achieves over 50% (i.e., 51.71%) BD-Rate reduction among all methods. The second best method obtains 49.68% BD-Rate gains, which is inferior to ours by 2.03% BD-Rate.

[1] https://groups.google.com/g/tensorflow-compression/c/LQtTAo6l26U/m/mxP-VWPdAgAJ

[2] https://r0k.us/graphics/kodak/

[3] https://tecnick.com/?aiocp\%20dp=testimages

[4] http://compression.cc/

Method	MSE			MS-SSIM
	Kodak	Tecnick	CLIC2022	Kodak
Cheng2020	4.84%	6.15%	8.29%	-44.00%
Entroformer	3.52%	2.04%	4.72%	-43.73%
NeuralSyntax	0.22%	-	-	-
QResVAE	-0.29%	1.02%	-0.13%	-
GLLMM	-2.64%	-5.66%	-	-48.32%
STF	-3.58%	-2.54%	-1.62%	-48.77%
WACNN	-4.05%	-	-	-48.86%
ELIC	-6.59%	-	-	-45.09%
Contextformer	-6.92%	-9.47%	-	-46.66%
Mixed	-11.12%	-11.79%	-	-49.68%
Ours	**-9.83%**	**-11.00%**	**-10.05%**	**-51.71%**

Table 1: Average BD-Rate (%) reduction against VVC anchor in different datasets

Method	Params	MACs (per pixel)	Enc.	Dec.
Cheng2020	26.5M	926k	1.9855s	4.0475s
Entroformer	44.9M	492k	-	-
NeuralSyntax	52.9M	7980k	-	-
QResVAE	34.0M	354k	0.1450s	0.0551s
GLLMM	-	-	-	-
STF	99.8M	508k	0.1296s	0.1242s
WACNN	75.2M	743k	0.1148s	0.1148s
ELIC*	33.8M	833k	0.1716s	0.0830s
Contextformer*	-	-	40s	44s
Mixed	75.8M	1781k	0.1405s	0.1300 s
Ours	78.8M	721k	0.1788s	0.1012s

Test Conditions: Intel i9-13900K CPU, Nvidia 4090 GPU, Windows 10. The enc./dec. time is averaged over all 24 images in Kodak, including entropy enc./dec. time.
*: We reproduced ELIC (He et al. 2022) to calculate the runtime. The Enc. & Dec. time of Contextformer is picked from (Koyuncu et al. 2022), which was tested on an NVIDIA Titan RTX GPU, i9-10980XE CPU.
MACs (per pixel) is calculated in an end to end manner.

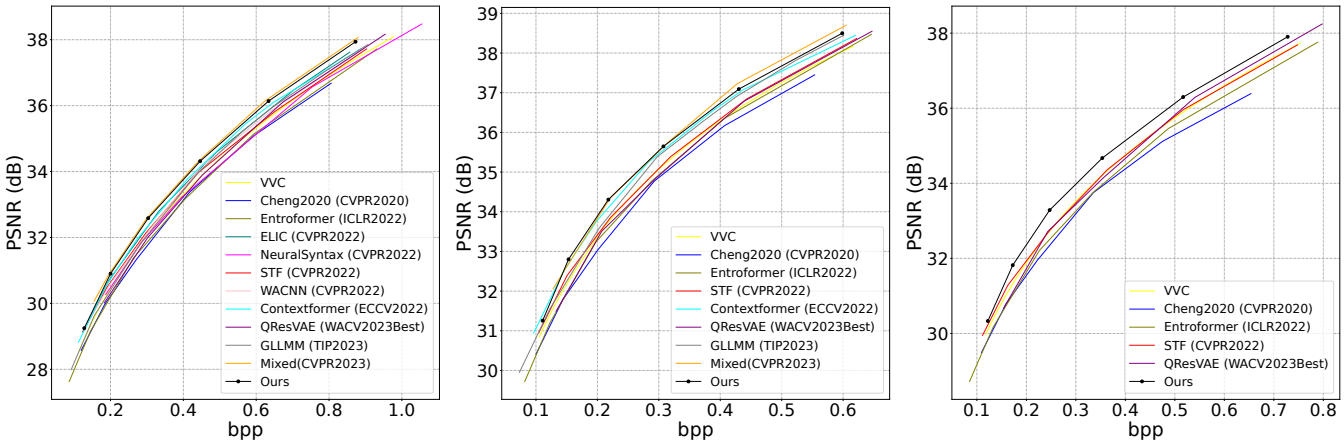
Table 2: Computational complexity compared with SOTAs

Qualitative Visualization

Figure 9 compares the qualitative results of our proposed CLIC and VVC. Here, the value of λ is set as 0.0018 in CLIC (corresponding to VTM-18.0 QP 42). As observed, CLIC yields more visually pleasing reconstructed images, exhibiting clearer textures and less noise.

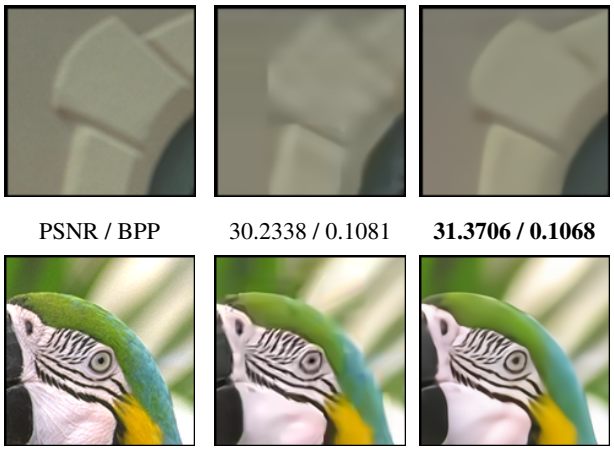
Complexity

Table 2 measures the computational complexity of each method using the number of parameters, MACs per pixel, encoding time (Enc.), and decoding time (Dec.). Our CLIC uses almost the same Context model as ELIC. Even though our number of parameters is higher than ELIC (78.8M vs. 33.8M), the computational complexity of MACs per pixel is significantly lower than ELIC (721K vs. 833K) due to the ef-



(a) Kodak (MSE) (b) Tecnick 1200x1200 (MSE) (c) CLIC 2022 (MSE)

Figure 8: RD curves of various methods. (a), (b), and (c) present MSE-optimized results. *Please zoom in for more details*.



PSNR / BPP	30.2338 / 0.1081	**31.3706 / 0.1068**
PSNR / BPP	31.6807 / 0.0635	**32.7820 / 0.0659**
(a) Original	(b) VTM-18	(c) Ours

Figure 9: Visualization of the reconstructed images from the Kodak dataset. Bold indicates the best performance.

ficiency of the clustering method. Our encoding and decoding time is also comparable with ELIC and slightly higher than other channel-conditional methods. Overall, our CLIC outperforms ELIC by a large margin, 9.83% vs. 6.59%, with almost the same complexity. In addition, Our MACs are only 40% of Mixed with a similar number of parameters (75.8M vs. 78.8M) and performance (1.29% BD-Rate lower on Kodak, 0.79% BD-Rate lower on Tecnick, 2.03% BD-Rate higher on MS-SSIM optimized Kodak).

Ablation Study

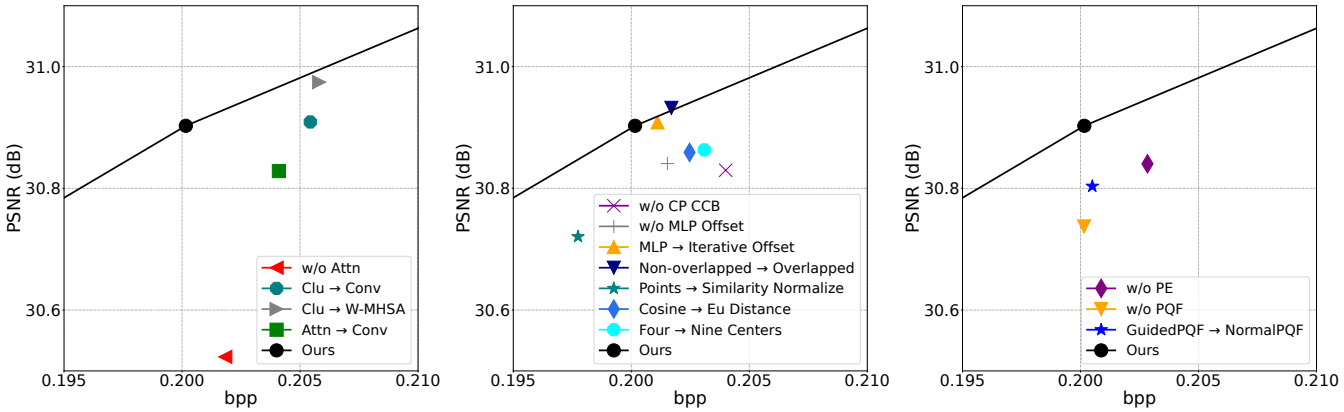
A series of ablation studies are conducted to verify the contribution of each modular component in CLIC.

Positional Encoding. Positional encoding (PE) plays a crucial role in CLIC. Figure 10(c) shows that "w/o PE" suffers from performance degradation. This occurs because pixel relationships are highly correlated with their positions: closer points exhibit higher correlation. Thus, we embed the PE module into CLIC for position-dependent processing.

Clustering in CCB. As illustrated in Figure 4, the clustering module (Clu for short) acts as a "Token Mixer" in CCB. We then implement the clustering module with other token mixers such as conventional CNN or Transformer for the same purpose. Specifically, we employ a 3×3 convolution layer or a Window-based Multi-Head Self-Attention (W-MHSA) (Liu et al. 2021) to replace the clustering module and remain other modules unchanged. Results reported in Figure 10(a) and Table 3 show that the use of W-MHSA causes a 5.14% increase in model parameters and a 1.94% reduction in MACs, whereas the use of convolutions increases 12.34% parameters and 25.02% MACs. Although the BD-Rate performance of using W-MHSA is comparable to ours, it greatly increases the encoding and decoding time by 67.35% and 127.49%. Due to the efficiency of the convolution-based implementation, its encoding/decoding time is reduced by only 3.49%/11.33%; however, its RD performance suffers.

Checkboard pattern in CCB. Besides, we apply a checkboard pattern at even-numbered CCBs. When this technique is removed, the coding efficiency dramatically suffers, shown as "w/o CP CCB" in Figure 10(b).

Attention in CCB. In each CCB, we introduce Spatial Attention and Channel Attention to augment the spatial and channel interaction on top of clustering features. The removal of these two attention units (w/o Attn) leads to significant performance degradation, as shown in Figure 10(a). In addition, when we replace both attention units with convolutions (see Table 3 and Figure 10(a)), the number of parameters, MACs, and coding/decoding time increase significantly, and the performance decreases dramatically. These results further demonstrate the validity of the proposed CCB, where the attention mechanism is effectively and efficiently combined with contextual clustering.



(a) CCB (b) Cluster Variants (c) GuidedPQF

Figure 10: Ablation experiments on modular components.

	Δ Parameters	Δ MACs	Δ Enc.	Δ Dec.
Clu → Conv	12.34%	25.02%	-3.49%	-11.33%
Clu → W-MHSA	5.14%	-1.94%	67.35%	127.49%
CA & SA → Conv	101.27%	59.93%	6.32%	1.91%
Ours	0%	0%	0%	0%

Table 3: Ablation experiments on Clustering in CCB

Clustering Variants. Next, we conduct ablation studies to discuss clustering-related methods. All results are provided in Figure 10(b).

A point worth exploring is overlapped versus non-overlapped clustering. In CLIC, each pixel point belongs to only one cluster. In addition, we use an overlapped clustering method where each point belongs to two clusters. Figure 10(b) shows that "Non-Overlapped → Overlapped" has little effect on the results while requiring extra complexity.

Besides, clustering centers also impact the results. We first examine the center offsets. It is observed that using MLP for center offsets yields comparable results to iteratively updating the centers (ten iterations), while "w/o MLP Offset" significantly degrades performance. As for the number of clustering centers, using nine clustering centers even performs worse than using four centers, which is probably because the involvement of more clusters makes the divisions too fine, isolating relevant points.

When clustering, to control the magnitude of F, we use $1 + m$ to scale it for normalization. Here, we implement another similarity-based normalization (Ma et al. 2023b), i.e., $1 + \sum_{i=1}^{m} \text{sigmoid}\,(\alpha \cdot s_i + \beta)$. Figure 10(b) demonstrates that the similarity-based normalization (Points → Similarity Normalize) yields inferior performance.

In addition, we use the Euclidean distance (Eu) to evaluate the similarity for clustering: each point is assigned to a center that has the minimum Euclidean distance to it. However, this method yields poor results because the Euclidean similarity only captures magnitude correlation while neglecting orientation and distribution similarity, thus limiting the correlation exploration across latent features.

GuidedPQF. We remove the proposed GuidedPQF and the corresponding coding performance is colored in orange in Figure 10(c). Moreover, we replace the GuidedPQF using a normal MSE-based PQF (Fu et al. 2023a), called NormalPQF in Figure 10(c). Clearly, both NormalPQF and GuidedPQF attain gains, and GuidedPQF gains more due to the guidance of the additional side information a_i^js.

Conclusion

In this paper, we present a Contextual Clustering based Learned Image Coding. We start by considering an image as a collection of individual pixel points and systematically reorganize all the points into several clusters via a clustering algorithm to exploit intra-cluster correlations. Then, we reorder all obtained point features according to their positions and further apply local attention mechanisms to the reordered features for inter-cluster correlation exploration. In this way, we achieve global feature characterization of an image, resulting in a more compact representation compared to conventional rectangular shape-based convolutions. Extensive experiments validate the superiority of our proposed method over state-of-the-art rule-based VVC and learned image compression methods.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (No. U20A20184 and 62171174).

References

Achanta, R.; Shaji, A.; Smith, K.; Lucchi, A.; Fua, P.; and Süsstrunk, S. 2012. SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11): 2274–2282.

Ba, J. L.; Kiros, J. R.; and Hinton, G. E. 2016. Layer Normalization. arXiv:1607.06450.

Ballé, J.; Laparra, V.; and Simoncelli, E. P. 2017. End-to-end Optimized Image Compression. *International Conference on Learning Representations*.

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

Bross, B.; Wang, Y.-K.; Ye, Y.; Liu, S.; Chen, J.; Sullivan, G. J.; and Ohm, J.-R. 2021. Overview of the Versatile Video Coding (VVC) Standard and its Applications. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(10): 3736–3764.

Bégaint, J.; Racapé, F.; Feltman, S.; and Pushparaja, A. 2020. CompressAI: a PyTorch library and evaluation platform for end-to-end compression research. arXiv:2011.03029.

Chadha, A.; and Andreopoulos, Y. 2021. Deep Perceptual Preprocessing for Video Coding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14852–14861.

Chen, F.; Xu, Y.; and Wang, L. 2022. Two-Stage Octave Residual Network for End-to-End Image Compression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 3922–3929.

Chen, T.; Liu, H.; Ma, Z.; Shen, Q.; Cao, X.; and Wang, Y. 2021. End-to-End Learnt Image Compression via Non-Local Attention Optimization and Improved Context Modeling. *IEEE Transactions on Image Processing*, 30: 3179–3191.

Cheng, Z.; Sun, H.; Takeuchi, M.; and Katto, J. 2020. Learned Image Compression with Discretized Gaussian Mixture Likelihoods and Attention Modules. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7939–7948.

Choi, Y.; El-Khamy, M.; and Lee, J. 2019. Variable rate deep image compression with a conditional autoencoder. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 3146–3154.

Ding, D.; Wang, J.; Zhen, G.; Mukherjee, D.; Joshi, U.; and Ma, Z. 2023. Neural Adaptive Loop Filtering For Video Coding: Exploring Multi-hypothesis Sample Refinement. *IEEE Transactions on Circuits and Systems for Video Technology*.

Duan, Z.; Lu, M.; Ma, Z.; and Zhu, F. 2023. Lossy Image Compression with Quantized Hierarchical VAEs. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 198–207.

Fu, H.; Liang, F.; Liang, J.; Li, B.; Zhang, G.; and Han, J. 2023a. Asymmetric Learned Image Compression with Multi-Scale Residual Block, Importance Scaling, and Post-Quantization Filtering. *IEEE Transactions on Circuits and Systems for Video Technology*.

Fu, H.; Liang, F.; Lin, J.; Li, B.; Akbari, M.; Liang, J.; Zhang, G.; Liu, D.; Tu, C.; and Han, J. 2023b. Learned Image Compression With Gaussian-Laplacian-Logistic Mixture Model and Concatenated Residual Modules. *IEEE Transactions on Image Processing*, 32: 2063–2076.

Gong, R.; Liu, X.; Jiang, S.; Li, T.; Hu, P.; Lin, J.; Yu, F.; and Yan, J. 2019. Differentiable soft quantization: Bridging full-precision and low-bit neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 4852–4861.

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

Guo-Hua, W.; Li, J.; Li, B.; and Lu, Y. 2023. EVC: Towards Real-Time Neural Image Compression with Mask Decay. In *International Conference on Learning Representations*.

He, D.; Yang, Z.; Peng, W.; Ma, R.; Qin, H.; and Wang, Y. 2022. ELIC: Efficient Learned Image Compression with Unevenly Grouped Space-Channel Contextual Adaptive Coding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5718–5727.

He, D.; Zheng, Y.; Sun, B.; Wang, Y.; and Qin, H. 2021. Checkerboard Context Model for Efficient Learned Image Compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14771–14780.

Hu, J.; Shen, L.; and Sun, G. 2018. Squeeze-and-excitation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7132–7141.

Hu, Y.; Yang, W.; and Liu, J. 2020. Coarse-to-Fine Hyper-Prior Modeling for Learned Image Compression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 11013–11020.

Kim, J.-H.; Heo, B.; and Lee, J.-S. 2022. Joint Global and Local Hierarchical Priors for Learned Image Compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5992–6001.

Kingma, D. P.; and Ba, J. 2014. Adam: A Method for Stochastic Optimization. arXiv:1412.6980.

Koyuncu, A. B.; Gao, H.; Boev, A.; Gaikov, G.; Alshina, E.; and Steinbach, E. 2022. Contextformer: A Transformer with Spatio-Channel Attention for Context Modeling in Learned Image Compression. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XIX*, 447–463. Springer.

Li, T.; Xu, M.; Zhu, C.; Yang, R.; Wang, Z.; and Guan, Z. 2019. A Deep Learning Approach for Multi-Frame In-Loop Filter of HEVC. *IEEE Transactions on Image Processing*, 28(11): 5663–5678.

Liu, J.; Liu, D.; Yang, W.; Xia, S.; Zhang, X.; and Dai, Y. 2020a. A Comprehensive Benchmark for Single Image Compression Artifacts Reduction. *IEEE Transactions on image processing*, 29: 7845–7860.

Liu, J.; Lu, G.; Hu, Z.; and Xu, D. 2020b. A Unified End-to-End Framework for Efficient Deep Image Compression. arXiv:2002.03370.

Liu, J.; Sun, H.; and Katto, J. 2023. Learned Image Compression with Mixed Transformer-CNN Architectures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14388–14397.

Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; and Guo, B. 2021. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10012–10022.

Lu, M.; Chen, F.; Pu, S.; and Ma, Z. 2022a. High-Efficiency Lossy Image Coding Through Adaptive Neighborhood Information Aggregation. arXiv:2204.11448.

Lu, M.; Guo, P.; Shi, H.; Cao, C.; and Ma, Z. 2022b. Transformer-based Image Compression. In *2022 Data Compression Conference (DCC)*, 469–469. IEEE.

Ma, C.; Wang, Z.; Liao, R.; and Ye, Y. 2021. A Cross Channel Context Model for Latents in Deep Image Compression. arXiv:2103.02884.

Ma, C.; Wu, Z.; Cai, C.; Zhang, P.; Wang, Y.; Zheng, L.; Chen, C.; and Zhou, Q. 2023a. Rate-Perception Optimized Preprocessing for Video Coding. arXiv:2301.10455.

Ma, X.; Zhou, Y.; Wang, H.; Qin, C.; Sun, B.; Liu, C.; and Fu, Y. 2023b. Image as Set of Points. In *International Conference on Learning Representations*.

Minnen, D.; Ballé, J.; and Toderici, G. D. 2018. Joint Autoregressive and Hierarchical Priors for Learned Image Compression. *Advances in Neural Information Processing Systems*, 31.

Minnen, D.; and Singh, S. 2020. Channel-wise Autoregressive Entropy Models for Learned Image Compression. In *2020 IEEE International Conference on Image Processing (ICIP)*, 3339–3343. IEEE.

Moon, Y. H.; Kim, G. Y.; and Kim, J. H. 2005. An efficient decoding of CAVLC in H. 264/AVC video coding standard. *IEEE Transactions on Consumer Electronics*, 51(3): 933–938.

Qian, Y.; Lin, M.; Sun, X.; Tan, Z.; and Jin, R. 2022. Entroformer: A Transformer-based Entropy Model for Learned Image Compression. In *International Conference on Learning Representations*.

Qian, Y.; Tan, Z.; Sun, X.; Lin, M.; Li, D.; Sun, Z.; Hao, L.; and Jin, R. 2021. Learning Accurate Entropy Model with Global Reference for Image Compression. In *International Conference on Learning Representations*.

Sullivan, G. J.; Ohm, J.-R.; Han, W.-J.; and Wiegand, T. 2012. Overview of the High Efficiency Video Coding (HEVC) Standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12): 1649–1668.

Sze, V.; and Budagavi, M. 2012. High Throughput CABAC Entropy Coding in HEVC. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12): 1778–1791.

Tang, Z.; Wang, H.; Yi, X.; Zhang, Y.; Kwong, S.; and Kuo, C.-C. J. 2023. Joint Graph Attention and Asymmetric Convolutional Neural Network for Deep Image Compression. *IEEE Transactions on Circuits and Systems for Video Technology*, 33(1): 421–433.

Wallace, G. K. 1992. The JPEG still picture compression standard. *IEEE Transactions on Consumer Electronics*, 38(1): xviii–xxxiv.

Wang, D.; Yang, W.; Hu, Y.; and Liu, J. 2022. Neural Data-Dependent Transform for Learned Image Compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 17379–17388.

Woo, S.; Park, J.; Lee, J.-Y.; and Kweon, I. S. 2018. CBAM: Convolutional Block Attention Module. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 3–19.

Yu, W.; Luo, M.; Zhou, P.; Si, C.; Zhou, Y.; Wang, X.; Feng, J.; and Yan, S. 2022. MetaFormer Is Actually What You Need for Vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10819–10829.

Zhang, M.; Deng, W.; and Li, X. 2022. A Unified Image Preprocessing Framework For Image Compression. arXiv:2208.07110.

Zhang, Y.; Ding, G.; Ding, D.; Ma, Z.; and Li, Z. 2024. On Content-Aware Post-Processing: Adapting Statistically Learned Models to Dynamic Content. *ACM Transactions on Multimedia Computing, Communications and Applications*, 20(1): 1–23.

Zhu, X.; Hu, H.; Lin, S.; and Dai, J. 2019. Deformable ConvNets v2: More Deformable, Better Results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9308–9316.

Zhu, Y.; Yang, Y.; and Cohen, T. 2022. Transformer-based transform coding. In *International Conference on Learning Representations*.

Zou, R.; Song, C.; and Zhang, Z. 2022. The Devil Is in the Details: Window-based Attention for Image Compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 17492–17501.