AdaNIC: Towards Practical Neural Image Compression via Dynamic Transform Routing

Lyfang Tao\textsuperscript{1,2,*}, Wei Gao\textsuperscript{1†}, Ge Li\textsuperscript{1}, Chenhao Zhang\textsuperscript{1}
\textsuperscript{1} SECE, Shenzhen Graduate School, Peking University,
\textsuperscript{2} Tencent AI Lab
luistao@tencent.com gaowei262@pku.edu.cn geli@ece.pku.edu.cn chenhaozhang@stu.pku.edu.cn

Abstract

Compressive autoencoders (CAEs) play an important role in deep learning-based image compression, but large-scale CAEs are computationally expensive. We propose a framework with three techniques to enable efficient CAE-based image coding: 1) Spatially-adaptive convolution and normalization operators enable block-wise nonlinear transform to spend FLOPs unevenly across the image to be compressed, according to a transform capacity map. 2) Just-unpenalized model capacity (JUMC) optimizes the transform capacity of each CAE block via rate-distortion-complexity optimization, finding the optimal capacity for the source image content. 3) A lightweight routing agent model predicts the transform capacity map for the CAEs by approximating JUMC targets. By activating the best-sized sub-CAE inside the slimmable supernet, our approach achieves up to 40% computational speed-up with minimal BD-Rate increase, validating its ability to save computational resources in a content-aware manner.

1. Introduction

In recent years, neural image compression (NIC) is being actively investigated, which reveals its great potential in terms of compression efficiency and capacity for perceptual optimization \cite{2, 3, 6, 24}. After initial attempts, the specific variants of autoencoders, namely compressive autoencoders (CAEs), have become a popular architecture choice in follow-up studies. The adoption of CAE for learning compact nonlinear representation of image signals leads to great success, yielding comparable or superior rate-distortion trade-offs when compared with the existing state-of-the-art codecs. Due to the learning-based nature of NIC, the number of incurred floating-point operations (FLOPs) is higher than those of legacy algorithms by orders of magnitude. By replacing traditional linear transforms (i.e., DCT, DST) with neural network-based nonlinear transforms, the inference computational costs will be huge although with much better representation capacity. Such a dilemma hinders the practical deployment of NIC codecs, which calls for an efficient way to reduce the computational overhead of CAEs without harming their performance advantages.

Early works have demonstrated that the scale of CAEs is highly related to the image quality or bitrate \cite{3}. The more radical quality objective in loss function will demand more latent channels allocated. Therefore, the converged model with inadequate channels will suffer from rate-distortion degradation. A larger redundant model carries no penalty or reward in terms of rate-distortion criteria. In this case, the well-studied channel pruning methods may fit the needs for complexity-mitigation. However, since neural image codecs are originally trained with diversified picture and block content and involve distortion-sensitive reconstruction, the contribution of each channel takes effect on individual inputs. Henceforth, when channel pruning approaches are applied to remove unimportant channels \cite{12, 23, 41}, excessive channel elimination can lead to severely-degraded rate-distortion performance. Therefore, the static way of one-shot channel pruning may not be suitable for further rate-distortion-complexity optimization, which can be crucial to the coding performance. Conversely, we would like to investigate the dynamic routing solution to benefit the underexplored rate-distortion-complexity (RDC) trade-off-oriented optimization.

In this paper, we emphasize the importance of employing content-adaptive optimization at run-time. The overall framework of AdaNIC is illustrated in Figure 1. By designing and training a lightweight adaptive routing agent under the rate-distortion lossless objective, and proposing...
the spatially-adaptive signal transform operators for fine-grained transform-capacity allocation, the optimal sub-CAE with minimal redundant parameters and computations can serve the corresponding input patches. In this way, the maximal system throughput can be achieved.

Since the action space of dynamic routing is devised as sample or region-adaptive, it can be seamlessly integrated into other feasible solutions for accelerating neural nonlinear transform that results in a static lightweight model and improves their performance by joint optimization. The rationale behind this is that the speed-up effects of AdaNIC come from exploiting the differences of required transform capacity among uncompressed content, which is not conflicting with efforts of acquiring a computationally-efficient model. The interesting side of the proposed “routing” approach is that it makes coding decisions at run-time, which is similar to the traditional RDO process or fast algorithms that are commonly adopted by modern image/video coding standards. Such kind of run-time trade-offs can bring about more flexibility when a codec system is responsible for the processing of a wide variety of content, enabling better rate-distortion or complexity tradeoffs through customized behaviors.

The contributions of our work can be summarized as follow: (1) a novel way of accelerating neural image codecs with content-adaptive transform routing (2) definition of unpenalized rate-distortion objective (3) network design of lightweight routing agent and its learning mechanism (4) an original spatially-adaptive convolution operator for fine-grained capacity allocation. The proposed solution utilizes a novel action space, which is content-adaptive (in image level or patch level), so additional optimization techniques (for example, model pruning / AutoML methods) shall be compatible with the trade-offs discussed in this work.

2. Related Work

2.1. Development of Image Compression

Compression technology is crucial for the production, storage, and transmission of digital multimedia assets. Some standardized compression methods for still image compression include JPEG [26], PNG [27], JPEG2000 [28], WebP [22], and HEIF [18].

Encouraged by the success of deep learning methods in various aspects of science and technology, the problem of learned image compression soon attract the interest of the research community. By substituting the engineered prediction, quantization or entropy coding modules with neural networks [7, 21, 30, 34], the capabilities of individual steps can be improved. Moreover, deep neural networks (DNNs) can be applied to image quality assessment [31] and enhancement by pre-processing [10] and post-processing [9].

Initially, recurrent neural networks (RNNs) are applied to extract image representations in an iterative (residual-based) way [16, 25, 33]. However, the approach of iterative inference is considered inefficient, despite its benefits for spatially-adaptive processing and variable rate.

Later, CNN-based CAEs [2, 32] succeed in achieving bet-
ter performance-efficiency trade-offs. Some key designs of CNN-based CAEs include the hyperprior branch [3] that transmits performance-sensitive side information via auxiliary bitstream, the generalized divisive normalization (GDN) layer [1] for noise-free normalization, and non-linearity.

To make continuous improvements for the compression performance, modern techniques such as visual attention mechanism, non-local architecture, residual learning, autoregressive (AR) CNN model for context modeling, probabilistic mixture model (e.g. GMMs) for entropy modeling, and so forth [5, 6, 24]. Among them, the AR context model is well-known as a sweet burden – it not only boosts the compression performance by a good margin but also significantly lengthens the coding and decoding process, as its data dependencies strictly prohibit parallel acceleration at run-time.

2.2. Practical Neural Image Compression

The sub-optimal processing efficiency of CAE-based NIC codecs restricts their practical usage and massive deployment. To decouple the indivisible parameter set and multiple rate-distortion objectives, some works [8, 32, 36] implement variable-rate codecs by feature modulation. Now that the inference of a full network is consistently required, the coding process for lower quality levels could be extended.

Taking advantage of the concept of slimmable neural networks, a specific CAE variant with variable transform capacity, namely SlimCAE, is introduced. Following the aforementioned rule, Yang et al. [35] propose an algorithm to search for the optimal (maximum) quality objective bound to the specific sub-CAE, where quality coefficients gradually reduce until relative RD performance stops improving.

The computational complexity also plays a critical role in the deployment of NIC codecs. In [15], model compression techniques are utilized to search for efficient CAE architectures, which is conducted by adding a weighted group LASSO regularization term concerning model FLOPs. Moreover, parallel-friendly architecture for context modeling is proposed to break down the path dependence of pixel-level AR modeling and alleviate latency caused by waiting. In [11], He et al. build a checkerboard-styled context model, which accelerates the generation of entropy parameters by applying a two-pass decoding pipeline. Both approaches focus on the delivery of an efficient static model, taking no advantage of content-adaptive computation.

2.3. Comparison

To highlight the varying motivation and effects, we compare the performance relative to independent CAEs (including time/space complexity and rate-distortion performance)

<table>
<thead>
<tr>
<th>Method</th>
<th>Motivation</th>
<th>Ref. Perf. (Independent CAEs as the anchor)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faster Codec</td>
<td>AdaNIC</td>
<td>-</td>
</tr>
<tr>
<td>SlimCAE</td>
<td>SAFT [29]</td>
<td>Near lossless (managed)</td>
</tr>
</tbody>
</table>

Table 1. Comparison between the proposed method and related work.

As shown in Table 1, previous works share the capability of reducing space complexity (storage consumption) of neural image compression by devising a single model that adapts to multiple quality levels (or bitrate targets). The introduced penalty in terms of time complexity and compression efficiency varies from severe to mild, according to their specific designs. In contrast, the proposed method aims at improving the overall throughput and puts no effort into reducing the number of independent models. Meanwhile, the proposed method barely introduces any additional storage overhead, which is guaranteed by our underlying architecture – a slimmable neural network [38, 39] with shared parameters.

3. Methods

3.1. Overview

As illustrated in Figure 1, the proposed AdaNIC framework is established as follows. First of all, we achieve dynamic capacity of neural transform units based on network slimming. Afterward, we extend the definition of existing slimmable operators, to support fine-grained control of spatially-adaptive transform capacity, which is the core component of the proposed SA-CAE module. Then, to gain optimal capacity control for the SA-CAE, a rate-distortion degradation tolerance-guided approach of capacity downgrading is devised as just-unpenalized model capacity (JUMC). Finally, an efficient yet powerful learning machine is proposed to capture the resulting rate-distortion characteristics of SA-CAE at different capacity levels and learn to make optimal routing decisions based on its learned representations.

3.2. Dynamic Transform Capacity

The neural transform unit is composed of a 2D signal convolution layer and a consecutive GDN layer, where the heavy computational burden is imposed to dense feature maps by massive feature extraction operations.

We first look into the rate-distortion performance of a typical CAE architecture, i.e., the mean-scale hyperprior CAE introduced in [24]. Following the paradigm of Slim-
CAE [35], we build up a slimmable supernet, whose activated channel number can be adjusted at run-time. By testing the coding performance of sub-models, we find that the observed degradation of coding performance caused by CAE capacity downgrading also heavily depends on different picture samples and texture patches, as depicted in Figure 2. To summarize, diversified coding performance and varying responses to transform capacity downgrading can be witnessed. The importance of gaining adaptive control of model capacity can be noted. If a static NIC codec is optimized to process all the inputs indiscriminately, the computational efficiency and coding performance will not easily get an optimal state for most input data, since their underlying differences in capacity demand are ignored. To overcome this deficiency, we propose to implement dynamic model capacity (TCM) which describes the designated spatial distribution of the output channel number. Correspondingly, the SA-GDN operator takes its role of nonlinear normalization based on the valid channel numbers at different spatial locations. An example of a neural transform unit established by the modified operators is shown in Figure 3. The inverse transform operations are handled by a similar module equipped with the SA-TConv operator with an equivalent definition. Assuming no stride and padding is applied, the output feature map \( F \in \mathbb{R}^{TCM(x,y) \times H \times W} \) can be computed with a fixed-sized input feature map \( X \in \mathbb{R}^{C \times H \times W} \):

\[
F(m, x, y) = \sum_{n=1}^{C} \sum_{u=-k}^{k} \sum_{v=-k}^{k} W(m, n, u, v) \cdot X(n, x-u, y-v),
\]

where \( W \in \mathbb{R}^{C' \times C \times K \times K} \) is the weight tensor of the SA-Conv tensor. \( C' \) represents the maximum number of filters supported by the operator.

To achieve real acceleration on existing hardware, block-based TCM is adopted for efficient implementation. If the granularity of TCM is very fine (e.g. pixel-level), the overhead of memory access (i.e. addressing) and limited choice for on-device convolution algorithms may restrict the conversion of reduced FLOPs to inference speedup. Following the block-based policy, the computation of 2D convolution is evolved by the proposed tiling-based partition mechanism. First, the input feature map is cropped to smaller pieces by the specific scheme shown in Figure 4, which guarantees the minimal occurrence of overlapped pixels and optimal efficiency. Afterward, the divided feature maps are individually processed to generate convolution results. Finally, intermediate results are merged to reconstruct a full-sized output feature map. The aforementioned strategy is developed by analysis of the sliding-window rule of convolution and is inspired by the underlying design of many AI accelerators [20].

Following the common practice of using a fixed filter number across layers in CAE, the TCM for the \( i^{th} \) stage is up-sampled from the next stage on the encoder side, keep-
ing the capacity allocation invariant in the direction of network depth. The $2 \times$ up-sampling is applied to align the dimension with the stride-2 convolution:

$$TCM^{(i)}(x, y) = TCM^{(i+1)}(\frac{x}{2}, \frac{y}{2}),$$  
(2)

and that $C$ in Eq. (1) can be replaced by:

$$\tilde{C} = \min(C, TCM(x, y)).$$  
(3)

The local performance of a block can get affected by its neighboring blocks under only one circumstance, where the up or the left-sided block assigned with fewer output channels gets processed by spatially-adaptive operators in the preceding layer. In this case, the reduced information on the edges of the input feature map may have a slight impact on the results, which are limited to pixels near the two borders.

### 3.4. Find Just-Unpenalized Model Capacity

Traditionally, the loss function of CAE is given as the joint rate-distortion objective:

$$\mathcal{L} = \lambda \times \mathcal{D} + \mathcal{R},$$  
(4)

where the distortion term $\mathcal{D}$ is evaluated by the $L_2$ loss function for objective quality, and the rate term $\mathcal{R}$ is estimated based on variational inference. When the balancing factor $\lambda$ is fixed, the optimality of the codec can be evaluated by the loss value.

As shown in Figure 2, the capacity downgrade of transform units results in various kinds of outcomes. Hence, evaluation of potential downgrading options is a necessary yet challenging problem. On one hand, the shifting on the curve cannot be scored solely by the distance metric, as the direction of the delta vector also plays an important role in its effects. On the other hand, although the rate and distortion metrics after downgrade can be considered in the neighborhood of those achieved at the highest capacity, the slope alone cannot reflect the degree of loss in terms of compression performance. Inspired by Eq. (4), the cost of a downgrade is defined as:

$$\mathcal{D}' = \frac{\lambda'}{\mu} \times \mathcal{D}$$  
(5)

$$\Delta \text{Cost} = \mathcal{L}'_{\text{new}} - \mathcal{L}'_{\text{old}} = \mu \times (\mathcal{D}'_{\text{new}} - \mathcal{D}'_{\text{old}}) + \mathcal{R}_{\text{new}} - \mathcal{R}_{\text{old}},$$  
(6)

where $\mu$ is the new balancing factor used when considering the rate-distortion trade-offs under the scenario of capacity routing, which can be within a certain range centering $\mu_0 = 1$. The distortion criterion in Eq. (6) is firstly normalized by the balancing factor for routing cost $\lambda'$ assigned for computing $\mathcal{L}'$, which can take a different value than $\lambda$ in Eq. (4). Note that Eq. (6) is established when $\mathcal{L}'$, $\mathcal{D}$, $\mathcal{R}$ are evaluated on a fine-grained basis, which means picture / block-level rate-distortion data are used instead of dataset-averaged results. By adding a tolerance threshold $\varepsilon$ to the original objective, the maximum tolerance of achieved coding loss $\mathcal{L}'_{\text{max}}$ is defined as:

$$\mathcal{L}'_{\text{max}} = \mathcal{L}'_{\text{old}} + \varepsilon.$$  
(7)

Following Eq. 7, the constrained rate-distortion results for the routing targets can be viewed as the area under the red line in Figure 5. Based on similar triangles, we have:

$$\mu = \tan \theta, d = \frac{\varepsilon}{\mu} \cos \theta,$$

$$\varepsilon = d \mu \sec \theta = d \sqrt{1 + \mu^2}.$$  
(8)

By setting the balancing factor $\mu$ (or the angle of $\theta$ as depicted in Figure 5) and the extrapolating distance $d$ to the coding cost line $\mathcal{L}'_{\text{old}}$ before routing, the threshold of added coding cost is thereby determined. Inspired by the concept of just-noticeable distortion (JND) [37] and just-recognizable distortion (JRD) [40], among all the candidate routing targets that fit in the designated range of negligible coding penalty, the sub-model capacity with the minimum
computational overhead becomes the desired action of the dynamic transform routing. Defined by Eq. (9), the label of expected action $z_{true}$ is named as just-unpenalized model capacity (JUMC):

$$z_{true} = \min(W), W = \{w|L'_w \leq L'_\text{max}\}, \quad (9)$$

where the capacity $W$ is indicated by the maximum number of filters in the main auto-encoder branch. $L'_w$ is the coding cost evaluated with capacity $w$. During inference, JUMC serves as TCM elements to control the scale of SA-CAE. The computational complexity is minimized because the “number of filters” - “FLOPs / inference latency” function is monotonically increasing for an SA-CAE.

The $w$ choice is limited to multiple fixed levels to simplify the label generation and ease the burden of paired learning models. Binary search algorithms are adopted to further accelerate the labeling process. If a fine-grained list of $w$ is supported, a binary search with limited steps of iteration can offer an approximate solution to the optimal routing target.

3.5. Transform Routing Agent Subsystem

As depicted in Figure 1, the training of a dynamic routing agent involves a two-stage learning pipeline. In the first stage, a large-scale learning model is employed to capture the rate-distortion responses of the SA-CAE model for various input signals, at all supported transform capacity levels. Thereafter, a lightweight student model is participated to learn by approximating the behavior of the pre-trained teacher model. The lightweight architecture of the student model can make the overhead of online routing decision generation close to negligible, hence facilitating its deployment in real-world coding workload.

The architecture of the teacher model can be summarized as two patch-level predictor branches built on top of a shared backbone $F_{bb}$. The backbone is a CNN-based feature extractor based on stacked inverted-residual blocks [14]. Two prediction heads are placed to handle different learning objectives. For the routing head, a set of target JUMC labels $z_{true}$ is generated by evaluating Eq. (9). For the $\Delta \text{Cost}$ head, extracted features are made use of to learn the mapping to the downwgrade cost vector:

$$F_{rt}: F_{bb}(x) \rightarrow z_{pred},$$
$$F_{dc}: F_{bb}(x) \rightarrow \Delta \text{Cost}_{pred}. \quad (10)$$

$F_{rt}$ and $F_{dc}$ are similarly constructed with two consecutive fully-connected layers, with a hard-swish activation function inserted in the middle. The parameters of $F_{rt}$ and $F_{dc}$ are optimized with the action loss function $\mathcal{A}$ and the surrogate loss function $\mathcal{S}$, respectively:

$$\mathcal{A}(z_{pred}, z_{true}) = \text{CrossEntropy}(z_{pred}, z_{true}),$$

$$\mathcal{S}(\Delta \text{Cost}_{pred}, \Delta \text{Cost}_{true}) = \frac{1}{\gamma} \text{MSE}(\Delta \text{Cost}_{pred} - \Delta \text{Cost}_{true}). \quad (11)$$

The combined loss function $L$ for the teacher model is formulated as:

$$L = \gamma \times A + (1 - \gamma) \times S. \quad (12)$$

where $\gamma$ is the balancing factor of the two correlated learning objectives. To control the scale of the student model, a lightweight student model with a minimized backbone and a single routing head is customized as in Figure 1 (b), to support online decision-making under a stricter resource constraint. The distillation loss function for the student model is given as:

$$D(z_{pred}, z_{teacher}, z_{true}) = \sigma \times K L(z_{pred}, z_{teacher}) + (1 - \sigma) \times A(z_{pred}, z_{true}), \quad (13)$$

where $KL$ stands for the Kullback-Leibler divergence [17], and $\sigma$ is the weighting factor that control the usage of distillation targets.

4. Experiments

4.1. Implementation Details

The proposed methods, as well as the representative methods for comparison, are implemented based on PyTorch deep learning framework. We use a combination of the CLIC professional and mobile training sets for model training. The Kodak dataset and the combined CLIC validation set are reserved for verification.

For the AdaNIC codec, multiple supersets targeting different quality levels (and bitrates) are independently initialized with corresponding architecture presets. The architecture-criterion mapping is shown in Table 2.

<table>
<thead>
<tr>
<th>Level</th>
<th>$q = 1$</th>
<th>$q = 2$</th>
<th>$q = 3$</th>
<th>$q = 4$</th>
<th>$q = 5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
<td>[96, 112, 128, 144, 160]</td>
<td>[128, 144, 160, 176, 192]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$M$</td>
<td>[128, 160, 192, 224, 256]</td>
<td>[192, 224, 256, 288, 320]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\lambda$</td>
<td>$1.2 \times 10^2$</td>
<td>$4.4 \times 10^{2}$</td>
<td>$1.6 \times 10^{3}$</td>
<td>$6.1 \times 10^{4}$</td>
<td>$1.2 \times 10^{5}$</td>
</tr>
</tbody>
</table>

Table 2. Architecture and optimization hyper-parameters of the proposed SA-CAE supernet in detail. A set of $\lambda$ coefficients are manually assigned to control the approximate range for the resulting bitrate. $N$ and $M$ denote the channel number of the main branch and the channel number of the hyperprior branch, respectively. Each quality level $q$ corresponds to a dedicated supernet, embedded with 5 capacity levels (C1-C5, from low to high) differentiated by corresponding $N$ and $M$ numbers.

The training process lasts for 1.6M iteration with a batch size of 8, taking about four days per quality level for the supernet, and one day for the teacher & student RA, with
two NVIDIA Tesla V100 GPUs. Considering the continuing use of a deployed codec system, the additional training cost of RA subsystem is deemed negligible.

4.2. Results

The evaluated rate-distortion-complexity results are given in Table 3. For each row in the table, individual supernet for the first 4 quality levels (q=1,2,3,4) are tested. The Bjøntegaard-delta-rate (BD-rate) is computed based on curve fitting as introduced in [4], which reflects the comprehensive coding performance (compression efficiency) at the concerned range of bitrates. The data regarding inference speed provided in the table are obtained by averaging results from different quality levels. The last two rows represent the coding performance achieved with the ground truth label $z_{true}$ and the labels generated by the routing agent $z_{pred}$, respectively. In Figures 6 - 8, the visualization of rate-distortion characteristics, inference latency, and the relationship between the two aspects are provided.

As it can be summarized from Table 3, the proposed routing agent can generally produce an additional speedup of 10%-25% for both CPU / GPU platforms over uniform slimming, at better or comparable compression performance. The BD-Rate increase is strictly-limited to within 1.0%, which means the achieved rate-distortion results can be considered as unpunished when compared to the sub-CAE with the highest capacity. The success in controlling the potential rate-distortion penalty highlights the value of the JUMC concept as proposed in Section 3.4.

By comparing the proposed method to DS-Net [19], which is a SOTA implementation of dynamic width for high-level vision tasks, the results are presented in Table 4. In can be observed that the proposed solution outperforms picture-level approaches in terms of speed-up, highlighting the effects of fine-grained control of model capacity.

4.3. Visual Quality

In Figure 9, two patches with the lowest and highest $z_{true}$ values are illustrated for comparison. Based on similar visualization efforts conducted on a large scale, it can be summarized that plain regions are generally more friendly to a CAE model with lower transform capacity, while image patches with intricate texture tend to have little room for capacity downgrade. We can also learn from picture that although the image below requires $z_{true} = 192$ to avoid significant RD degradation in assessed by objective metrics, the image reconstructed with the sub-CAE with lowest dynamic transform capability still have fair visual quality. It hints that criteria that more closely-related to the results of subjective quality assessment should be used in future work, to better eliminate perceptual redundancy on a fine-grained basis.

4.4. Ablation Study

The effectiveness of the proposed architecture for routing agent is demonstrated by performing ablation studies. First, we investigate various approaches that capable of reducing the inference overhead of the routing agent model, including decreasing input resolution, and using the lightweight architecture. The acceleration results are reported in Table 5. We also present the predictive performance gain achieved by adopting the proposed double-headed architecture for the teacher model, and the single-
Table 3. Overview of coding performance and inference time complexity of the proposed method. The SA-CAE supernets at their highest capacity are chosen as the anchors for rate-distortion performance and inference latency data in the table. “SA-CAE @ CY” indicates the uniform adoption of y-th capacity of the supernet.


Table 6. Predictive performance of the proposed routing agents under different ablation settings. “Routing” and “ΔCost” stands for the routing head and the cost head. “KD” on/off correspond to σ = 0.5/0.0 in Eq. (13). “Hi-res” denotes the input resolution of 256 × 256. “Acc.†”, “Deg. †”, and “MAE” correspond to decision accuracy, proportion of routing decisions exceeding JUMC (which causes undesired rate-distortion degradation), and the mean-absolute-error between the predicted decision and the JUMC labels.

5. Conclusion

In this paper, we present a novel way for the flexible model-capacity control for CAEs. The comprehensive solution includes a set of novel spatially-adaptive operators, an optimal capacity assignment algorithm based on degradation cost thresholding, and a learning system for dynamic transform routing, which is lightweight yet robust. The neural-based transform is thereby streamlined with the guidance from an online-generated capacity map, and additional convolution filters can be applied only to blocks where they are profitable. The superior experiment results reveal that a new perspective of joint rate-distortion-complexity optimization for neural image compression has been established by acknowledging and predicting the differences in terms of coding efficiency and signal-transform capacity requirements across images and patches.

References


[22] Li Lian and Wei Shilei. Webp: A new image compression format based on vp8 encoding. Microcontrollers & Embedded Systems, 3, 2012. 2


