Learning Image and Video Compression through Spatial-Temporal Energy Compaction

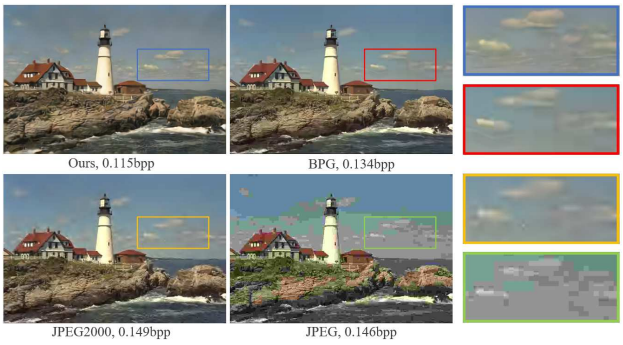
Zhengxue Cheng, Heming Sun, Masaru Takeuchi, Jiro Katto
Department of Computer Science and Communication Engineering, Waseda University, Tokyo, Japan
{zxcheng@asagi., hemingsun@aoni., masaru-t@aoni., katto@}waseda.jp

Abstract

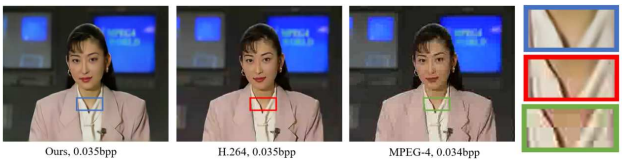
Compression has been an important research topic for many decades, to produce a significant impact on data transmission and storage. Recent advances have shown a great potential of learning image and video compression. Inspired from related works, in this paper, we present an image compression architecture using a convolutional autoencoder, and then generalize image compression to video compression, by adding an interpolation loop into both encoder and decoder sides. Our basic idea is to realize spatial-temporal energy compaction in learning image and video compression. Thereby, we propose to add a spatial energy compaction-based penalty into loss function, to achieve higher image compression performance. Furthermore, based on temporal energy distribution, we propose to select the number of frames in one interpolation loop, adapting to the motion characteristics of video contents. Experimental results demonstrate that our proposed image compression outperforms the latest image compression standard with MS-SSIM quality metric, and provides higher performance compared with state-of-the-art learning compression methods at high bit rates, which benefits from our spatial energy compaction approach. Meanwhile, our proposed video compression approach with temporal energy compaction can significantly outperform MPEG-4 and is competitive with commonly used H.264. Both our image and video compression can produce more visually pleasant results than traditional standards.

1. Introduction

Data compression has been a significant research topic in the field of signal processing for several decades. In terms of image compression codecs, existing image compression standards, such as JPEG [1], JPEG2000 [2], and BPG, which uses the intra-coded HEVC [3], rely on the hand-crafted encoder-decoder pipeline. These image formats are widely used in various image applications. Conventional



(a) Reconstructed images *kodim21* from kodak dataset.



(b) Reconstructed frame *akiyo_cif* from VTL dataset.

Figure 1: Visualized results of our approach and commonly used image/video compression standards. With the same bit bugets, our approach produces more visually pleasant results than conventional standards.

video coding algorithms, including H.261, MPEG-4 Part 2, commonly used standard H.264/AVC [4], most recent standard HEVC/H.265 [3], have also achieved impressive performance through efforts spanning several decades. However, along with the proliferation of high-resolution images and videos, as well as the development of novel image/video formats, existing standards are not expected to be the optimal compression solution for all types of contents.

Recently, deep learning has been successfully applied to compression tasks. There are several potential advantages for learning compression to enhance the performance of image and video compression. First, the encoder-decoder pipeline in conventional compression standards resembles an autoencoder to learn high-level representation. Although

autoencoders are basically applied to dimensionality reduction tasks [5], they are able to achieve better compression performance. Most recent learning based compression approaches, including recurrent neural networks [8, 9, 10], convolutional neural networks [11, 12, 13, 14, 15, 16, 17] and generative adversarial networks [18, 19, 20], have all adopted the autoencoder architecture. Next, the temporal redundancy of video compression can be intuitively reduced by using learning based video prediction, generation and interpolation approaches. The works [21, 22] have already achieved promising results by using prediction or interpolation neural networks. The last advantage of learning based compression is that, although the development and standardization of a conventional codecs has historically spanned several years, a deep learning based compression approach can be adapted much quicker because all of the parameters in an autoencoder architecture can be learned in an automatic and unsupervised manner. Therefore, learning compression is expected to become more generalized and more efficient.

However, there are still issues to be addressed. Generally speaking, image compression exploits spatial distribution, and video compression exploits temporal distribution to learn high-level features. Good spatial-temporal energy compaction is important for high coding efficiency, according to traditional digital coding theory [25]. Previous works use rate-distortion optimization, but few works analyze whether spatial-temporal energy is well-compacted or not.

In this paper, we present a convolutional autoencoder architecture with differential quantization and entropy estimation for image compression. Thereby, we propose to add an energy compaction-based penalty into loss function, to achieve higher image compression performance. Furthermore, we generalize our image compression to video compression, by adding an interpolation loop to image encoder and propose to adaptively select number of frames in one interpolation loop by analyzing temporal energy distribution.

We compare our image compression to state-of-the-art image standards and recent learning approaches. Our approach achieves significantly better MS-SSIM in comparison with the latest image compression standard BPG and outperforms state-of-the-art learning compression methods at high bit rate. On the other hand, our video compression is competitive with H.264 with MS-SSIM and produce more visually pleasant reconstructed videos.

2. Related Work

Hand-crafted Compression Existing image compression standards, such as JPEG [1], JPEG2000 [2], and BPG, which uses the intra-coded HEVC [3], reply on hand-crafted module design individually. Specifically, these modules include intra prediction, discrete cosine transform or wavelet transform, quantization, and entropy coder such as Huffman coder or content adaptive binary arithmetic coder (CABAC). They design each module with multiple modes and conduct the rate-distortion optimization to determine the best mode. During the development of next-generation compression algorithms, some hybrid methods have been proposed to improve the performance, by taking advantage of both conventional compression algorithms and latest machine learning approaches, such as [6, 7].

State-of-the-art video compression algorithms, such as HEVC/H.265 [3], H.264 [4], MPEG-4 Part 2, incorporate the inter prediction into the encoder architecture. Inter prediction utilize the temporal similarity of neighboring frames to reduce the transmitted information. As for the order of reference frames, both H.264 and HEVC/H.265 support two configurations, that is *low delay P* and *random access*. *low delay P* only use the previous frames as uni-directional references, while *random access* allows bidirectional referencing in a hierarchical way. *Random access* can achieve higher coding efficiency than *low delay P*. The key technique in inter prediction is integer and fractional motion estimation using block matching and motion compensation.

Learning Compression Recently, end-to-end image compression has attracted great attention. Some approaches proposed to use recurrent neural networks (RNNs) to encode the residual information between the raw image and the reconstructed images in several iterations, such as the work [8, 9] optimized by mean-squared error (MSE) or the work [10] optimized by MS-SSIM [30]. Some generative adversarial networks (GANs) based techniques are proposed in [18, 19, 20] for high subjective reconstruction quality at extremely low bit rates. Other notable approaches include differentiable approximations of round-based quantization [11, 12, 14] for end-to-end training, content-aware importance map [17], hyperprior entropy model [13] and conditional probability models [15] for entropy estimation.

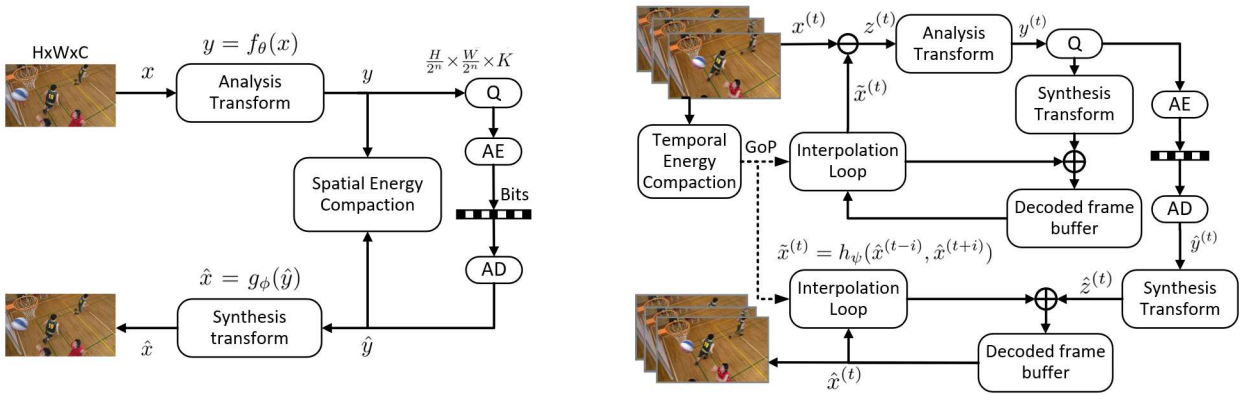
However, learning video compression has not yet been largely exploited. Only a few related works [21][22] have been proposed. Wu *et al.* [21] firstly proposed to use image interpolation network to predict frames except for key frames. Chen *et al.* [22] relied on traditional block based architecture to use CNN nets for predictive and residual signals. It is highly desired to further exploit learning video compression algorithms.

3. Proposed Method

Our proposed learning image and video compression is illustrated in Fig. 2, and image compression models serve for frames in video compression architecture.

3.1. Learning Image Compression

Given an image, a compression system can be considered as an analysis transform f at the encoder side, and a synthesis transform g at the decoder side, as shown in



(a) Learning Image Compression

(b) Learning Video Compression

Figure 2: Overview of our proposed learning image and video compression with spatial-temporal energy compaction.

Fig. 2(a),

$$y = f_\theta(x)$$
$$\hat{x} = g_\phi(\hat{y}) \tag{1}$$

where x, $\hat{x}$, y, and $\hat{y}$ are the original images, reconstructed images, compressed data (also called latent presentation) before quantization, and quantized compressed data, respectively. θ and ϕ are optimized parameters in the analysis and synthesis transforms, respectively.

To obtain high-level features, the analysis and synthesis transforms can be composed into a sequence of consecutive down(up)sampling operations, which can be implemented by convolutional or transposed convolutional filter with a stride of 2. Our network architecture mainly refer to the autoencoder in [12], but according to [24], it is pointed that super resolution is achieved more efficiently by first convolving the images and then upsampling, instead of first upsampling and then convolving. Thus we use 2 convolutional filters as one down(up)sampling unit, and the network architecture is given in Fig. 3. Assume we have n downsampling units and the number of convolutional filters in the last layer is K, the compressed data y will have the dimension of $\frac{H}{2^n} \times \frac{W}{2^n} \times K$. In practice, $n = 3$, $K = 48$, H and W are set as 128 due to memory limitation.

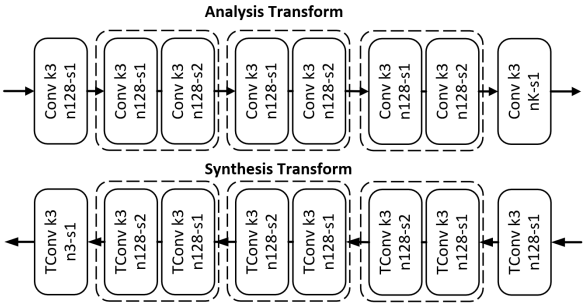


Figure 3: Network architecture of analysis and synthesis transform, where "k3 n128-s2" represents a convolution layer with kernel size 3, 128 channels and a stride of 2. T-conv represents transposed convolutional layers.

Based on the rate-distortion cost function in traditional codecs, the loss function is defined as follows:

$$J(\theta, \phi; x) = \lambda D(x, \hat{x}) + R(\hat{y}) \tag{2}$$

where λ controls the tradeoff between the rate and distortion. D represents the distortion between the original images x and reconstructed images $\hat{x}$; R represents the bits required to encode the quantized compressed data $\hat{y}$.

3.1.1 Quantization and Entropy Estimation

In Fig. 2(a), Q represents quantization, AE and AD represent the arithmetic encoder and arithmetic decoder, respectively. In traditional codecs, quantization is implemented by using a round function (denoted as $Q[\cdot]$), and its derivative is almost zero except at the integers. Therefore, it cannot be directly incorporated into the gradient-based optimization process. Several quantization approximations have been proposed, such as uniform noise approximation [12] and soft vector quantization [14]. The approximate quantized $\hat{y}$ are shown in Fig. 4, where soft vector quantization has a shaping parameter σ and high σ leads to accurate results, low σ is good for smooth gradient propagation. In other studies [11][15], the derivation was replaced in the back propagation only, but it was guaranteed that the quantized value was correct in the forward propagation. By conducting experiments, we found different quantization methods have very little effect on the compression performance. For simplicity, we used additive uniform noise approximation.

According to the Shannon theory [34], the rate is lower-bounded by the entropy of the discrete probability distribu-

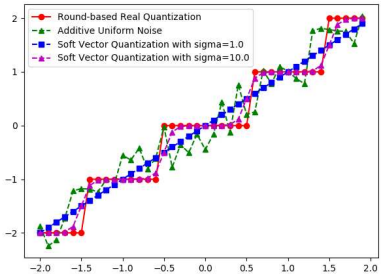


Figure 4: Performance with different quantization methods.

tion of the quantized codes, as follows:

$$R = \mathop{\mathbb{E}}_{\tilde{y}\sim q}\left[-\log_2(p_{\tilde{y}}(\tilde{y}))\right] \tag{3}$$

where q is the actual distribution of the compressed code $\tilde{y}$ and $p_{\tilde{y}}(\tilde{y})$ is the entropy model. Thus, several entropy estimation methods have been introduced by related studies, including soft histogram based entropy estimation [14], non-parametric factorized entropy model [13], 3D-CNN based conditional probability model [15], and hyperprior based entropy model [13]. We used a fully factorized entropy model [13], which yields promising image compression performance. Factorized entropy model produces an estimated entropy and serves for AE and AD. During the test, we can use JPEG2000 entropy coder to generate bitstream.

3.1.2 Spatial Energy Compaction Constraint

According to digital coding theories [25], good energy compaction property is critical for high coding efficiency performance. The analysis transform converts input x into compressed data y with K spatial channels, which resembles a subband coding system. In a subband coding system, for any arbitrary transforms (which is not required to be non-orthogonal), the energy of K spatial channels satisfies [26],

$$\begin{aligned} \sigma_y^2 &= A_k \sigma_x^2 \\ \sigma_r^2 &= \sum_{k=0}^{K-1} B_k \sigma_q^2 \end{aligned} \tag{4}$$

where q is the quantization error for each spatial channel, i.e. $q = \hat{y} - y$, and r is reconstructed error of images, i.e. $r = \hat{x} - x$. σ^2 denotes the variance of a certain data to represent the energy. A_k describes the energy distribution of channels in the analysis transform, determined by x and optimized parameter θ; B_k measures the extent of the quantization error's impact on the reconstructed error for a specified channel in the synthesis transform, determined by quantization errors and parameter ϕ. A_k can be easily calculated by obtaining the variance of x and y. B_k is determined by both the quantization errors and the parameter ϕ during the synthesis transform. B_k can be estimated by constructing several fake compressed data c_k, whose k-th channel is all-1 and other channels are all-0. By feeding these c_k as $\hat{y}$ into a given pre-trained synthesis transform individually, we stack $\sigma_{\hat{x}}^2$ to form B_k. Both A_k and B_k have the dimension of $K \times 1$.

Based on Eq.(4), optimum bit allocation is formulated [26]. With a constant rate constraint, minimized reconstruction error is given by

$$\min\{\sigma_r^2\} \propto \prod_{i}^{K-1} (A_k B_k) \tag{5}$$

Detailed proof can be referred to [26]. If $\prod_i^{K-1}(A_k B_k)$ can be minimized, spatially energy can be optimally compact. Thus, we propose to add a penalty in loss function to regularize A_k and B_k. First, we need to center the energy in a few channels as much as possible. We normalize A_k by dividing its sum, therefore, normalized A_k measures the energy distribution for y_k. For example, if $A_k[e] = 0.8$ for the e-th channel, 80% of energy will be distributed in the e-th channel. Then, we construct a penalty term by using the entropy of the energy distribution as follows:

$$P = \mathbb{E}[-A_k \log_2 A_k] \tag{6}$$

We add this penalty to the loss function. After several iterations, most of the energy is centered only in one or a few channels, while the other channels have little energy. We denote the channel with the largest energy as e. Next, we minimize the corresponding $B_k[e]$ to make $A_k B_k$ minimized, and the penalty term is defined as:

$$P = B_k[e] \tag{7}$$

Finally, the loss function is defined as:

$$J(\theta, \phi; x) = \lambda D(x, \hat{x}) + R(\hat{y}) + \beta P(A_k, B_k) \tag{8}$$

where β controls the influence of the penalty term on the loss function.

3.2. Learning Video Compression

Considering that a video consists of consecutive frames, a video compression system can be simply extended from image compression system as

$$\begin{aligned} y^{(t)} &= f_\theta(x^{(t)}) \\ \hat{x}^{(t)} &= g_\phi(\hat{y}^{(t)}) \end{aligned} \tag{9}$$

where $x^{(t)}$ represents $\{x^{(0)}, x^{(1)}, \cdots, x^{(T)}\}$, and similar notations are for $y^{(t)}, \hat{y}^{(t)}, \hat{x}^{(t)}, t \in [0, T)$. We define T as group of pictures (GOPs) as HEVC/H.265 [3], which can be encoded and decoded independently. Two consecutive groups share the same boundary frames.

Due to the temporal similarity of neighboring frames, encoding residual information between two frames can gain more coding efficiency than encoding them separately. Thus, a more general form of video compression system is rewritten as

$$\begin{aligned} z^{(t)} &= x^{(t)} - \tilde{x}_E^{(t)} \\ y^{(t)} &= f_\theta(z^{(t)}) \\ \hat{z}^{(t)} &= g_\phi(\hat{y}^{(t)}) \\ \hat{x}^{(t)} &= \hat{z}^{(t)} + \tilde{x}_D^{(t)} \end{aligned} \tag{10}$$

where $\tilde{x}_E^{(t)}$ and $\tilde{x}_D^{(t)}$ are predicted frame using neighboring frames at encoder and decoder side, repectively. For the first frame, there is no previous information, i.e. $\tilde{x}_E^{(0)} = \tilde{x}_D^{(0)} = 0$, video compression reduces to a image compression, therefore, we call these key frames as I-frame. The block diagram of the proposed learning image compression is illustrated in Fig. 2(b).

3.2.1 Interpolation Loop

The closer generated predictive frame $\tilde{x}^{(t)}$ gets to raw frames $x^{(t)}$, the fewer information $z^{(t)}$ has. Therefore, a high-quality frame interpolation network is desired. We use the latest work [35] to formulate the interpolation as a convolution h over two neighboring frames as

$$\tilde{x}^{(t)} = h_\psi(x^{(t-i)}, x^{(t+i)}) \tag{11}$$

where i is the distance between reference frames and generated frames. ψ are optimized parameters in interpolation network h. More importantly, according to Eq.(10), predicted frames should be kept the same at both encoder and decoder side to prevent the quality gap,

$$\tilde{x}_E^{(t)} = \tilde{x}_D^{(t)} \tag{12}$$

Therefore, the encoder and decoder should see the same information equally. Then, the input for interpolation network should be reconstructed frames, not raw frames. Eq.(11) is rewritten as

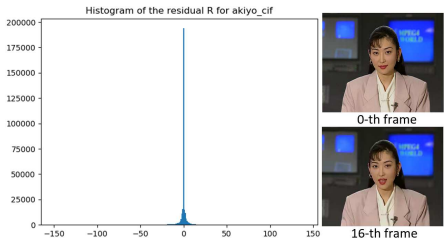
$$\tilde{x}^{(t)} = h_\psi(\hat{x}^{(t-i)}, \hat{x}^{(t+i)}) \tag{13}$$

We use a local interpolation loop at the encoder side to store reconstructed frames in the buffer, as Fig. 2(b).

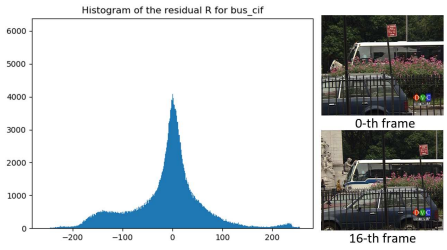
3.2.2 Temporal Energy Compaction

To further reduce the temporal redundancy, inspired by HEVC random access [3] and the work [21], we use a hierarchy interpolation method, which can be illustrated as

$$\begin{aligned} & z^{(0)} = x^{(0)}, z^{(T)} = x^{(T)} \\ & \tilde{x}^{(\frac{T}{2})} = h_\psi(\hat{x}^{(0)}, \hat{x}^{(T)}) \\ & \tilde{x}^{(\frac{T}{4})} = h_\psi(\hat{x}^{(0)}, \hat{x}^{(\frac{T}{2})}),\ \tilde{x}^{(\frac{3*T}{4})} = h_\psi(\hat{x}^{(\frac{T}{2})}, \hat{x}^{(T)}) \end{aligned} \tag{14}$$



(a) Sequence *Akiyo* in VTL, H_T=2.673.



(b) Sequence *Bus* in VTL, H_T=7.999

Figure 5: Examples of Temporal Energy Histogram for R_T

The hierarchy interpolation is recursively conducted until all the frames are reconstructed.

Each video contents has different motion textures, so the T should be adaptively selected to fit the motion characteristic of videos, so we propose an adaptive T determination strategy based on temporal energy compaction. We define the temporal motion difference in two neighboring I-frames with a proper distance τ (in our experiments $\tau = 16$) as

$$R_T = x^{(\tau)} - x^{(0)} \tag{15}$$

then, we consider the distribution of R_T, and calculate the entropy of R_t as

$$H_T = \mathbb{E}[-P_{R_T} \log_2 P_{R_T}] \tag{16}$$

H_T describes the motion characteristic of video sequences, as shown in Fig. 5. Large H_T implies that this video has fast motion objects, while low motion videos has small H_T. Then we propose to select the T using

$$T = \begin{cases} 2, & U \le H_T \\ 8, & L \le H_T < U \\ 16, & H_T < L \end{cases} \tag{17}$$

where L, U are constants for lower and upper bounds. Low motion videos are assigned with $T = 16$, that is, intermediate $(T-1)$ frames can be interpolated, without destroying the quality. In this case, $z^{(t)}$ is already small, so we send fewer I-frames to achieves temporal energy compaction. As for high motion videos, T is only set as 2, because I-frames do not provide enough information to interpolate a high-quality frame, so we remove the hierarchy interpolation to prevent the error propagation.

4. Implementation Details

Dataset To train our image compression models, we used a subset of ImageNet database [28], and cropped them into millions of 128×128 samples. For testing, we used commonly used Kodak lossless image database [29] with 24 uncompressed 768×512 images. To validate the robustness of our proposed method, we also tested the proposed method on the CVPR workshop CLIC validation dataset [23] with large and various resolutions up to about 2K. To test the performance of video compression, we use the widely used Video Trace Library (VTL) dataset [36], which includes 20 videos with the resolution of 352×288 and 8 test sequences with the resolution of 832×480 and 416×240, which are commonly used by video coding standardization group with rich texture scenes and motion scenes.

Training Details To train our image compression autoencoder, the model was optimized using Adam [27] with a batch size of 16. The learning rate was maintained at a fixed value of 1×10^{-4} during the training process. In the Eq.(8), β was set to 0.001. In our experiments, we add the energy compaction penalty to the model at high bit rate, and train for several 10^5 iterations, and then train the model up to several 10^6 iterations for each λ. By introducing different λ to fine-tune a pre-trained autoencoder, we can obtain variable bit rates. We have found that by changing λ with a pre-trained autoencoder, the energy distribution property will not be changed largely, as long as the initial state of the parameters in the models already has a good spatially energy compaction. Thus, we only consider the penalty for one λ trial when training the neural network. Here, we obtained six models with λ in the set $\{2, 4, 8, 16, 32, 64\}$.

In our video compression approach, we use the pre-trained models of [35] to build our reconstruction loop. By checking the histogram of H_T using VTL dataset, we used $L = 6.0$, $U = 8.0$ in Eq.(17) to ensure majority of sequences to select proper T and averaged T is equal to 8.

Measurements To achieve high subjective quality, we used a popular MS-SSIM [30] as distortion term defined by $D = 1 - \text{MS-SSIM}(\mathrm{x}, \hat{\mathrm{x}})$. To measure the coding efficiency, the rate is measured in bits per pixel (bpp), and the rate-distortion (RD) curves are drawn to demonstrate their coding efficiency.

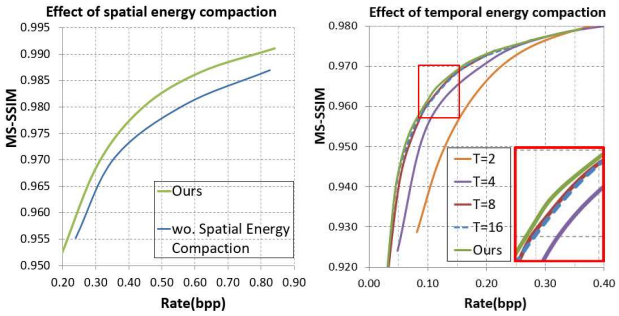
5. Experiments

In this section, we conduct experiments to evaluate the performance, and present comparison results.

5.1. Ablation study

In order to show the effectiveness of our proposed spatial-temporal energy compaction approach, we first perform the following ablation study.

We compare the performance of our image compression with spatial energy compaction constraint to the case without energy constraint. The RD performance averaged on the Kodak dataset is presented in Fig. 6(a). It is observed that energy compaction constraint can help autoencoder to gain a higher coding efficiency, especially with large bit budgets.



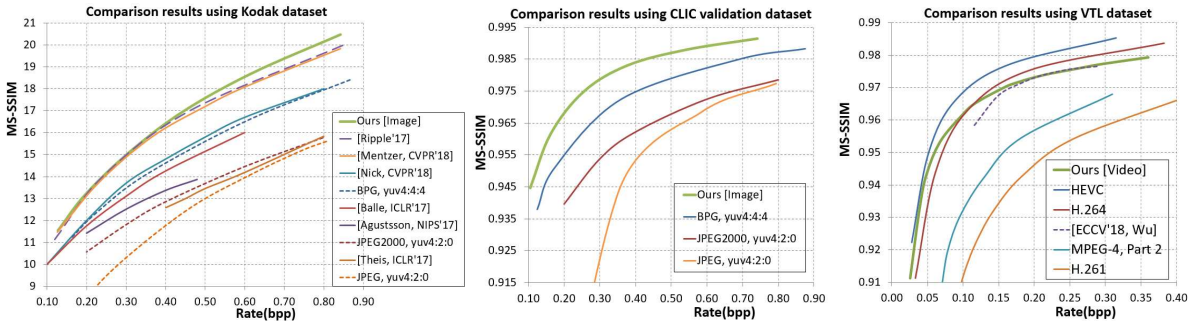
(a) MS-SSIM evaluated on Kodak. (b) MS-SSIM evaluated on VTL.

Figure 6: Ablation Study.

To visualize how temporal energy compact works, we conduct the experiments of learning video compression with different T, as shown in Fig. 6(b). Along with the increasing of T, coding efficiency gets improved, but the performance almost saturates between $T = 8$ and $T = 16$. Our approach can adaptively select T to achieve better rate-distortion optimization than the case with constant T.

5.2. Performance of learning image compression

We compare our method with well-known compression standards, and recent neural network-based learned compression methods, as shown in Fig. 7(a), where MS-SSIM is converted to decibels ($-10\log_{10}(1-\text{MS-SSIM})$) to illustrate the difference clearly. For JPEG and JPEG2000, we used the official software libjpeg [31] and OpenJPEG [32], which uses the default configuration *YUV420* format. The state-of-the-art image compression standard was BPG [33], for which we used the non-default *YUV444* format refer to [15][18]. Because the source codes of previous neural networks based approaches were not available, we carefully traced the point in the RD curves from the respective studies of Nick *et al.* [10], Theis *et al.* [11], Ballé *et al.* [12], and Ripple *et al.* [18]. The data in Mentzer *et al.* [15] were obtained from their project page. It can be observed that our method significantly outperforms Nick *et al.* [10], Theis *et al.* [11], Ballé *et al.* [12]. Our methodology is better than the work of Mentzer *et al.* [15] and Ripple *et al.* [18] at high bit rate, because our spatially energy compact constraint can allocate bits more efficiently, with higher bit budgets. Our proposal achieves comparable performance with Mentzers work *et al.* [15] and Ripple *et al.* [18] at low bit rate. Currently we only use a factorized entropy model and our method does not depend on the design of entropy mod-



(a) Kodak image dataset. (b) CLIC image dataset. (c) VTL video dataset.

Figure 7: Comparison results using different datasets.

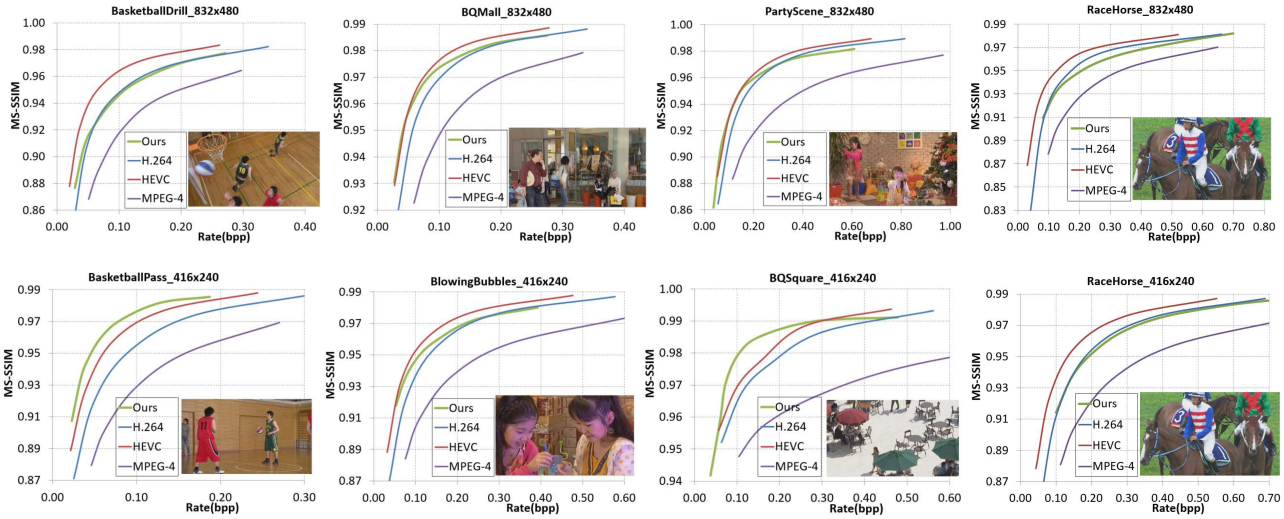


Figure 8: Comparison results for each video sequence.

els. Thus, in future work, our bit allocation method can be combined with a more complicated context adaptive entropy model, such as [13], to yield a better result. Fig. 7(b) shows the comparison between JPEG, JPEG2000, and BPG averaged on the CLIC validation dataset. Our method outperformed JPEG, JPEG2000 and BPG significantly in terms of MS-SSIM, with high resolution images.

5.3. Performance of learning video compression

We compare our learning video compression with state-of-the-art video compression algorithm and recent learning video compression methods [21]. For fair comparison, we use the the averaged results on each video sequences as [21]. The performance using VTL is shown in Fig. 7(c). For HEVC/H.265 and H.264, we use the official software HM 16.0 [37] and JM 19.0 [38] with random access configuration. The GOP is set as 8 and intra period is also 8 to make the comparison fair. For MPEG-4 Part2 and H.261, we use FFMPEG software. The data point of [21] are from their original paper. It is observed that our method outperforms MPEG-4 and H.261 significantly and is competitive with H.264. Moreover, we offer a wide range of bit rates and achieve better performance even at low bit rate than the work [21], which benefits from our proposed interpolation loop and temporal energy compaction.

To cover a variety of video contents, we also test our codec using common test sequences, following the work [22], whose results are a little worse than H.264. The RD curves of eight sequences are shown in Fig. 8. It can be observed that our method outperforms H.264 for most sequences, and even outperforms HEVC/H.265 for sequences *BasketballPass* and *BQSquare*.

5.4. Qualitative results

We visualize some reconstructed images and videos to demonstrate qualitative performance.

Figure 9: Example of one reconstruction image *kodim01* from Kodak dataset.

Figure 10: Example of one reconstruction frame in Video *paris_cif* from VTL dataset.

The reconstructed images are shown in Fig. 1(a) and Fig. 9. Fig. 1(a) shows examples *kodim21* with approximately 0.12 bpp and a compression ratio of 200:1. It can be observed that the cloud above the sea appear more natural in our reconstructed images using 0.115 bpp less bits than BPG, JPEG2000, and JPEG. Particularly, for the BPG-encoded images, blocking artifacts occur in the sky when a large compression ratio is applied. Fig. 9 shows examples *kodim01* under approximately 0.24 bpp with a compression ratio of 100:1, because the raw images are a lossless PNG format with 24 bpp (8 bit for each color component). Thus, it is observed that the latch on the door is maintained well in our reconstructed images. However, the images are blurry for the BPG, JPEG2000 and JPEG reconstructed images.

Some reconstructed frames from VTL dataset are shown in Fig. 1(b) and Fig. 10. Using the interpolation loop, the rate can be greatly saved. Clear block artifacts are observed for MPEG-4 compressed frames. Many details and shapes, such as woman's eyes in Fig. 10, are destroyed in H.264 compressed frames. Unlike them, our approach do not have any block artifacts to produce visually pleasant results.

6. Conclusion

We propose learning image and video compression approach through spatial-temporal energy compaction property. Specifically, we propose to add a spatial energy compaction-based penalty into loss function in image compression models, to achieve higher performance. We generalize image compression to video compression with an interpolation loop and adaptive interpolation period selection based on entropy of temporal information.

Experimental results demonstrate that our proposed image compression outperforms BPG with MS-SSIM quality metric, and provides higher performance compared with state-of-the-art learning compression methods. Our video compression approach outperforms MPEG-4, and is competitive with commonly used H.264. Both our image and video compression can produce more visually pleasant results than traditional standards.

References

[1] G. K Wallace, *"The JPEG still picture compression standard"*, IEEE Trans. on Consumer Electronics, vol. 38, no. 1, pp. 43-59, Feb. 1991.

[2] Majid Rabbani, Rajan Joshi, *"An overview of the JPEG2000 still image compression standard"*, ELSEVIER Signal Processing: Image Communication, vol. 17, no, 1, pp. 3-48, Jan. 2002.

[3] G. J. Sullivan, J. Ohm, W. Han and T. Wiegand, *"Overview of the High Efficiency Video Coding (HEVC) Standard"*, IEEE Transactions on Circuits and Systems for Video Technology, vol. 22, no. 12, pp. 1649-1668, Dec. 2012.

[4] T. Wiegand, G. J. Sullivan, G. Bjontegaard, A. Luthra, *"Overview of the H.264/AVC Video Coding Standard"*, IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, no. 7, pp. 560-576, July. 2003.

[5] P. Vincent, H. Larochelle, Y. Bengio and P.-A. Manzagol, *"Extracting and composing robust features with denoising autoencoders"*, Intl. conf. on Machine Learning (ICML), pp. 1096-1103, July 5-9. 2008.

[6] Z. Cheng, H. Sun, M. Takeuchi, J. Katto, *"Performance Comparison of Convolutional AutoEncoders, Generative Adversarial Networks and Super-Resolution for Image Compression"*, CVPR Workshop and Challenge on Learned Image Compression (CLIC), pp. 1-4, June 17-22, 2018.

[7] Z. Chen, Y. Li, F. Liu, Z. Liu, X. Pan, W. Sun, Y. Wang, Y. Zhou, H. Zhu, S. Liu, *"CNN-Optimized Image Compression with Uncertainty based Resource Allocation"*, CVPR Workshop and Challenge on Learned Image Compression (CLIC), pp. 1-4, June 17-22, 2018.

[8] G. Toderici, S. M.O'Malley, S. J. Hwang, et al., *"Variable rate image compression with recurrent neural networks"*, arXiv: 1511.06085, 2015.

[9] G, Toderici, D. Vincent, N. Johnson, et al., *"Full Resolution Image Compression with Recurrent Neural Networks"*, IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 1-9, July 21-26, 2017.

[10] Nick Johnson, Damien Vincent, David Minnen, et al., *"Improved Lossy Image Compression with Priming and Spatially Adaptive Bit Rates for Recurrent Networks"*, arXiv:1703.10114, pp. 1-9, March 2017.

[11] Lucas Theis, Wenzhe Shi, Andrew Cunninghan and Ferenc Huszar, *"Lossy Image Compression with Compressive Autoencoders"*, Intl. Conf. on Learning Representations (ICLR), pp. 1-19, April 24-26, 2017.

[12] J. Balle, Valero Laparra, Eero P. Simoncelli, *"End-to-End Optimized Image Compression"*, Intl. Conf. on Learning Representations (ICLR), pp. 1-27, April 24-26, 2017.

[13] Johannes Balle, D. Minnen, S. Singh, S. J. Hwang, N. Johnston, *"Variational Image Compression with a Hyperprior"*, Intl. Conf. on Learning Representations (ICLR), pp. 1-23, 2018. `https://tensorflow.github.io/compression/`

[14] E. Agustsson, F. Mentzer, M. Tschannen, L. Cavigelli, R. Timofte, L. Benini, L. V. Gool, *"Soft-to-Hard Vector Quantization for End-to-End Learning Compressible Representations"*, Neural Information Processing Systems (NIPS) 2017, arXiv:1704.00648v2.

[15] F. Mentzer, E. Agustsson, M. Tschannen, R. Timofte, L. V. Gool, *"Conditional Probability Models for Deep Image Compression"*, IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), June 17-22, 2018. `https://github.com/fab-jul/imgcomp-cvpr`

[16] Z. Cheng, H. Sun, M. Takeuchi, J. Katto, *"Deep Convolutional AutoEncoder-based Lossy Image Compression"*, Picture Coding Symposium, pp. 1-5, June 24-27, 2018.

[17] M. Li, W. Zuo, S. Gu, D. Zhao, D. Zhang, *"Learning Convolutional Networks for Content-weighted Image Compression"*, IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), June 17-22, 2018.

[18] Ripple Oren, L. Bourdev, *"Real Time Adaptive Image Compression"*, Proc. of Machine Learning Research, Vol. 70, pp. 2922-2930, 2017.

[19] S. Santurkar, D. Budden, N. Shavit, *"Generative Compression"*, Picture Coding Symposium, June 24-27, 2018.

[20] E. Agustsson, M. Tschannen, F. Mentzer, R. Timofte, and L. V. Gool, *"Generative Adversarial Networks for Extreme Learned Image Compression"*, arXiv:1804.02958.

[21] C-Y Wu, N. Singhal, P. Krahenbuhl, *"Video Compression through Image Interpolation"*, 15th European Conference on Computer Vision, September 8 C 14, 2018.

[22] T. Chen, H. Liu, Q. Shen, T. Yue, X. Cao, and Z. Ma. *"Deepcoder: A deep neural network based video compression"*. 2017 IEEE Visual Communications and Image Processing (VCIP), pp. 1C4, Dec 2017.

[23] Workshop and Challenge on Learned Image Compression, CVPR2018, `http://www.compression.cc/challenge/`

[24] W. Shi, J. Caballero, F. Huszar, et al. *"Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network"*, Intl. IEEE Conf. on Computer Vision and Pattern Recognition, June 26-July 1, 2016.

[25] N.S. Jayant and P. Noll, *"Digital coding of waveforms"*, Englewood Cliffs NJ, Prentice-Hall, 1984.

[26] J.Katto and Y.Yasuda: *"Performance Evaluation of Subband Coding and Optimization of Its Filter Coefficients"*, SPIE Visual Communication and Image Processing, Nov.1991.

[27] D. P. Kingma and J. Ba, *"Adam: A method for stochastic optimization"*, arXiv:1412.6980, pp.1-15, Dec. 2014.

[28] J. Deng, W. Dong, R. Socher, L. Li, K. Li and L. Fei-Fei, *"ImageNet: A Large-Scale Hierarchical Image Database"*, IEEE Conf. on Computer Vision and Pattern Recognition, pp. 1-8, June 20-25, 2009.

[29] Kodak Lossless True Color Image Suite, Download from http://r0k.us/graphics/kodak/

[30] Z. Wang, E. P. Simoncelli and A. C. Bovik, *"Multiscale structural similarity for image quality assessment"*, The 36-th Asilomar Conference on Signals, Systems and Computers, Vol.2, pp. 1398-1402, Nov. 2013.

[31] JPEG official software libjpeg, https://jpeg.org/jpeg/software.html

[32] JPEG2000 official software OpenJPEG, https://jpeg.org/jpeg2000/software.html

[33] BPG Image Format, https://bellard.org/bpg/

[34] C. E. Shannon, *"A Mathematical Theory of Communication"*, The Bell System Technical Journal, Vol. 27, pp. 379-423, July, 1948.

[35] S. Niklaus, L. Mai and F. Liu, *"Video Frame Interpolation via Adaptive Separable Convolution"*, IEEE International Conference on Computer Vision (ICCV) 2017.

[36] Video Trace Library. `http://trace.eas.asu.edu/index.html`

[37] K. McCann, C. Rosewarne, B. Bross, M. Naccari, K. Sharman, G. J. Sullivan, *"High Efficiency Video Coding (HEVC) Test Model 16 (HM 16) Encoder Description"*, Document JCTVC-R1002, Sapporo, Jul. 2014. `https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/`

[38] A. M.Tourapis, K. Suhring, G. Sullivan, *"H.264/14496-10 AVC Reference Software Manual"*, Document JVT-AE010, London, UK, 28 June- 3 July 2009. `http://iphome.hhi.de/suehring/tml/download/`