End-to-End Learning of Video Compression Using Spatio-Temporal Autoencoders

Jorge Pessoa
INESC-ID
Instituto Superior Técnico
Universidade de Lisboa
Lisboa, Portugal
jorge.pessoa@tecnico.ulisboa.pt

Helena Aidos
LASIGE
Faculdade de Ciências
Universidade de Lisboa
Lisboa, Portugal
haidos@fc.ul.pt

Pedro Tomás
INESC-ID
Instituto Superior Técnico
Universidade de Lisboa
Lisboa, Portugal
pedro.tomas@inesc-id.pt

Mário A. T. Figueiredo
Instituto de Telecomunicações
Instituto Superior Técnico
Universidade de Lisboa
Lisboa, Portugal
mario.figueiredo@lx.it.pt

***Abstract*—*Deep learning* (DL) is revolutionizing image and video processing and now holds state-of-the-art performance in many tasks. However, video compression has so far resisted the DL revolution. Current attempts rely on complex solutions, interconnecting multiple networks to mimic the different layers of conventional codecs. Since DL approaches usually excel when the models are allowed to learn their own feature set, a different solution is herein proposed: end-to-end learning of a single network, explicitly avoiding motion estimation/prediction. We formalize it as the rate-distortion optimization of a single spatio-temporal autoencoder, by jointly learning a latent-space projection transform, and a synthesis transform for low-bitrate video compression. The quantizer uses a rounding scheme, relaxed during training, and an entropy estimation technique to enforce an information bottleneck. The obtained video compression network shows competitive performance against standard codecs (MPEG-4 Part 2, H.264/AVC, H.265/HEVC), particularly for low bitrates, even while avoiding the use of any motion prediction/compensation method.**

***Index Terms*—Autoencoder, End-to-End Learning, Video Compression, Motion Prediction Free**

I. Introduction

The fast technological improvements in imaging devices lead to higher-resolution media, outpacing the increase in capacity to store and transfer data. Hence, there is a growing need for more efficient (preferably patent-free) compression schemes to expedite the process of sharing visual media.

The strong influence of deep learning (DL) coupled with the relevance of image compression has sparked a quest for DL-based approaches, which results are already on par with state-of-the-art codecs (*e.g.*, WebP and BPG) [1]. Different architectures have been proposed (e.g., [1]–[3]), generally based on a common principle: the original image is transformed into a latent representation by an encoder; this representation is quantized and entropy-coded; and a decoder reconstructs the image from the quantized latent representation. This type of approach relies on the ability of *deep neural networks* (DNN) to extract meaningful and compact representations from 2D data. Convolutional autoencoders are particularly suited to this task, due to their ability to exploit the structural redundancy present in images and learn efficient representations.

DL-based video compression methods typically follow standard codecs, by relying on motion estimation followed by residual compression [4]–[7]; or, similarly, by compressing key frames and performing interpolation-based reconstruction of the remaining (e.g., [8], [9]) or through block-based frame prediction combined with frame reconstruction (e.g., [10], [11]). However, these methods use complex multi-network architectures leading to high encoding times.

In this work, we explicitly depart from the traditional video coded structures, by avoiding any type of explicit motion prediction and tackling video compression via end-to-end optimization of a single spatio-temporal autoencoder architecture. We propose a rate-distortion optimization framework for video compression that, beyond learning a 3D latent-space representation of video, also enforces temporal consistency between frames, avoiding unwanted artifacts (e.g., flickering) in the resulting video sequence (contrasting with [12]). We perform end-to-end feature learning with a spatio-temporal autoencoder architecture that can be employed on high-resolution videos (in opposition to [13]), by optimizing a loss function that not only takes into consideration the reconstruction distortion, but also an estimate of the length of the entropy-coded quantized latent representation (bit-rate) and the temporal consistency between frames. The resulting video compression network, at low bitrates, is competitive with state-of-the-art codecs, namely the highly developed and optimized H.265/HEVC [14], the previous-generation H.264/AVC, and outperforms MPEG-4 Part 2. Hence, this article contributions can be summarized as: (i) a simple formulation of the video compression problem as the search for a latent-space representation, from which the video is reconstructed after quantization; (ii) an end-to-end rate-distortion video optimization framework that minimizes inconsistencies between frames; (iii) a 3D convolutional autoencoder architecture, with multi-scale connections applied to video compression, trained using the aforementioned optimization framework; and, (iv) a hyperprior network to capture spatio-temporal dependencies in the resulting latent-space representation.

This work was partially supported by Portuguese national funds through Fundação para a Ciência e a Tecnologia (FCT) under projects HAnDLE, ref. PTDC/EEI-HAC/30485/2017, ResNetDetect, ref. PCIF/MPG/0051/2018, the LASIGE Research Unit, ref. UIDB/00408/2020 and ref. UIDP/00408/2020, the INESC-ID Research Unit, ref. UIDB/50021/2020, and the Instituto de Telecomunicações Research Unit, ref. UIDB/50008/2020.

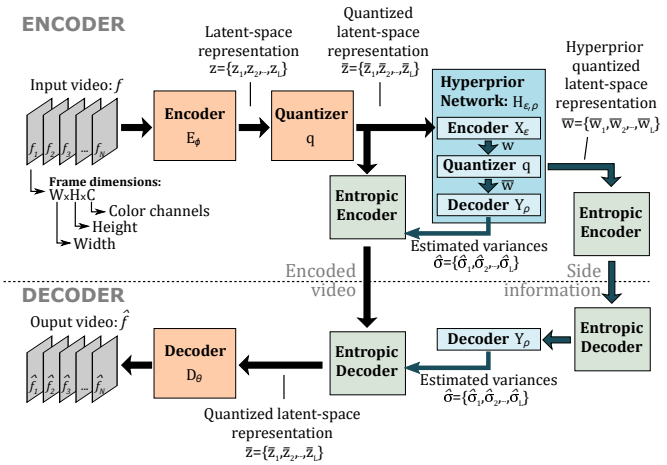

Fig. 1: Illustration of the video compression framework.

II. End-to-End Learning of Video Compression

The proposed compression architecture uses two transforms: a projection transform to represent the original video in a latent space, and a synthesis transform to reconstruct the video into the original space. The video is processed on a 3D pixel grid with one color channel (grayscale) or with multiple color channels (*e.g.*, three for the most common RGB, YUV or YCbCr spaces, or four for the CMYK space). The compressed video is the result of entropy-coding the quantized latent-space representation, and the reconstruction is obtained by applying the synthesis transform to that representation. The transforms and the quantizer are jointly learned on an end-to-end fashion.

A. Problem Formulation

Formally, a video $f = \{f_1, f_2, \ldots, f_N\} \in \mathcal{F}^N$ is a sequence of N frames, where $f_i \in \mathcal{F}$, with $\mathcal{F}$ denoting the space to which each frame belongs (*e.g.*, $[0; R]^{W\times H\times 3}$, for RGB, YUV or YCbCr frames with $(W \times H)$ pixels, and pixel values limited to the range $[0; R]$). As shown in Fig. 1, an encoder $\mathrm{E}_\phi : \mathcal{F}^N \to \mathcal{Z}$, where $\mathcal{Z} = \mathbb{R}^L$ is the latent space and L its dimensionality, extracts a representation $z = \mathrm{E}_\phi(f)$. The latent-space representation is then quantized, $\bar{z} = q(z)$, where $q : \mathcal{Z} \to \mathcal{S}$ is a quantizer, with some finite code-book $\mathcal{S}$. Finally, the quantized latent-space representation $\bar{z} \in \mathcal{S}$ is used by a decoder $\mathrm{D}_\theta : \mathcal{S} \to \mathcal{F}^N$ to reconstruct an approximation $\hat{f}$ of the original video f.

In this formulation, ϕ and θ are the parameters of the encoder (or projection transform) and the decoder (synthesis transform), respectively. The whole network $\mathrm{C}_{\phi,\theta} : \mathcal{F}^N \to \mathcal{F}^N$ is the composition of the encoder, quantizer, and decoder:

$$\hat{f} = \mathrm{C}_{\phi,\theta}(f) = \mathrm{D}_\theta\Big(q\big(\mathrm{E}_\phi(f)\big)\Big). \tag{1}$$

In this work the quantizer q represents quantization by rounding each component of z to the nearest integer, thus $\mathcal{S}$ is simply a finite subset of $\mathbb{Z}^L$ (vectors of integers).

The compressed representation of the video results from the entropic coding of $\bar{z} = q\left(\mathrm{E}_\phi(f)\right)$. Optimal entropic coding depends on the probability distribution of $\bar{z}$, $p_{\bar{z}}$, which must be estimated during training to allow rate-distortion optimization. For this, as in [1] we note that the components of $z = \mathrm{E}_\phi(f)$ are not independent. Hence, each element z_i is modeled as $\mathcal{N}(0, \sigma_i)$ and made dependent on other components of z (in a limited neighbourhood). These variances are estimated by an additional autoencoder, $\mathrm{H}_{\epsilon,\rho} : \mathcal{Z} \to \mathbb{R}_+^L$ (the *hyperprior* network), itself the composition of an encoder $\mathrm{X}_\epsilon : \mathcal{Z} \to \mathcal{W}$ (where $\mathcal{W}$ is the latent space of the hyperprior), the same rounding quantizer q as above, and a decoder $\mathrm{Y}_\rho : \mathcal{S} \to \mathbb{R}_+^L$. As the variances are needed to decode the entropy-coded bitstream, the quantized latent representation of the hyperprior, $\bar{w} = q(w) = q\big(\mathrm{X}_\epsilon(z)\big)$, must also be sent to the decoder as part of the compressed representation. Thus, the hyperprior can be seen as a side-information channel that improves the overall rate-distortion performance of the encoder, although contributing to the bitstream size.

To jointly optimize (in the rate-distortion sense) the main ($\mathrm{C}_{\phi,\theta}$) and the hypeprior ($\mathrm{H}_{\epsilon,\rho}$) networks, an entropy model for the hyperprior needs to be estimated during training. The probability distribution of $\bar{w}$ is modeled as a factorized model

$$p(\bar{w}|\delta) = \prod_i p(\bar{w}_i|\delta_i), \tag{2}$$

where each δ_i is estimated during training. Specifically, each δ_i is obtained by approximating $p(\bar{w}_i|\delta_i)$ by a probability density function (pdf). The pdf $p : \mathbb{R} \to \mathbb{R}^+$ in (2) is defined in terms of its cdf $P : \mathbb{R} \to [0, 1]$ ($P(x) = \int_{-\infty}^{x} p(y)dy$), as a general approximator of density functions [1]. The two functions are defined as follows, for $k = 1, \ldots, K$ (herein we set $K = 5$),

$$\begin{aligned} P &= g_K \circ g_{K-1} \circ \cdots \circ g_1, \\ p &= g'_K\, g'_{K-1} \cdots g'_1, \\ g_k(x) &= h_k(H^{(k)}x + b^{(k)}), \quad \text{for } k = 1, ..., K-1, \\ g_K(x) &= \mathrm{sigmoid}(H^{(k)}x + b^{(K)}), \\ h_k(x) &= x + a^{(k)} \odot \tanh(x), \end{aligned} \tag{3}$$

where $\circ$ represents the composition of functions, g' is the derivative of g, $H^{(k)}$ and $b^{(k)}$ are matrices and vectors, respectively, and $\odot$ denotes the Hadamard (component-wise) product. To simplify the model, we assume that $\delta_i = \delta_j$, if i and j are in the same hyperprior latent representation channel.

B. Network Architecture

To design an efficient architecture for both the encoder E_ϕ and decoder D_ϕ, both recursive and 3D Convolutional Neural Networks (CNN) were considered. However, on our tests, a 3D CNN architecture achieved a better performance, as it processes spatial and temporal information simultaneously, allowing a better information control when producing highly compressed video sequences. Also, such architecture does not impose any restrictions on the size of the input video sequence, as the compressed bitstream for frame i is only dependent on a fixed temporal neighbourhood of k adjacent frames (and not on the result of the coding process). Hence, one can process a sequence of n frames in a single iteration, or decompose it into multiple subsets of N_s frames. Moreover, it does not impose

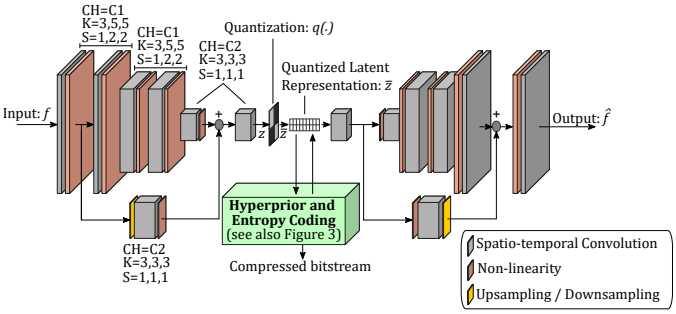

Fig. 2: Spatio-temporal autoencoder architecture for video compression. The parameters of each block correspond to the convolution configuration (CH – convolution channels; K – kernel size; S – stride). When the parameters are presented as a set of three numbers, the first refers to the temporal dimension and the remaining two to the spatial dimensions.

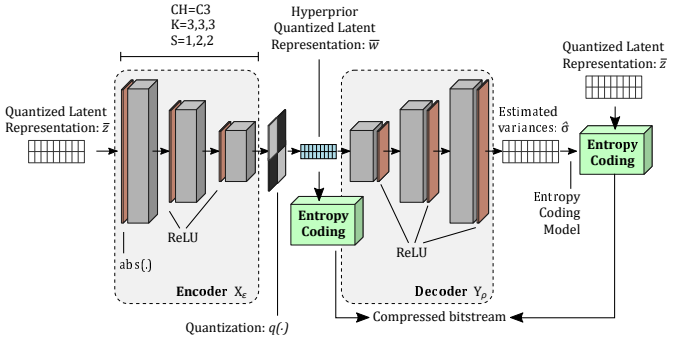

Fig. 3: Spatio-temporal autoencoder architecture for the hyperprior network (CH – convolution channels; K – kernel size; S – stride). When the parameters are presented as a set of three numbers, the first refers to the temporal dimension and the remaining two to the spatial dimensions.

any specific limits on the size of the set, nor that all sets must be equal sized. This is particularly interesting as it decreases the memory requirements for compression/decompression and also promotes parallel processing on heterogeneous devices and systems, thus decreasing computation time.

On the other hand, the frames in a video exhibit different correlations at different spatial scales. Hence, we use a dual-scale architecture (see Fig. 2) to better adapt to the variable dependency/redundancy scale of the input video (the use of additional scales showed marginal improvements that are statistically insignificant). Hence, the main path of the network is structured as a regular deep encoder, while the secondary path resembles a high-level skip connection.

The encoder E_ϕ is therefore composed of multiple processing blocks on two different scales (see Fig. 2), each including a 3D convolution and a non-linearity, expressed as leaky ReLUs (with a leak of 0.2). The latent representation results from adding the two referred scales and processing the result with a final inter-scale block. The shallower path uses trilinear downsampling to align both outputs. The decoder D_θ is built by reversing the encoder order of operations, and replacing convolutions by their transposes, and downsampling operations by the corresponding upsamplings.

To allow rate-distortion optimization, an hyperprior network is used to estimate the variance by taking into account both spatial and temporal redundancies in $\bar{z}$ (as inherited from z). Hence, we structure $\mathrm{H}_{\epsilon,\rho}$ as a spatio-temporal autoencoder, as shown in Fig. 3. The encoder X_ϵ uses three sequential processing blocks with a 3D convolution ($C3$ channels, kernel size 3 and stride (1,2,2)) and a ReLU non-linearity. The decoder Y_ρ inverts X_ϵ by replacing convolutions by the corresponding transposed convolutions.

C. Optimization Framework

Parameter learning cannot be carried out directly on the proposed structure, since the quantizer would block the back-propagation of the gradients through the bottleneck [1]–[3]. To circumvent this difficulty we exploit a classical result from quantization theory [15]: high-rate quantization noise/error is well approximated by additive noise, with uniform distribution in the quantization interval. Hence, during training, quantization is replaced by additive noise with a uniform density on the interval $[-\frac{1}{2}, \frac{1}{2})$, since the quantizer q simply rounds each component of its input to the nearest integer.

The proposed network is end-to-end optimized for both the reconstruction loss ($\mathcal{L}_r$) and the total entropy of the representation ($\mathcal{L}_h$). Additionally, we include a third loss term encouraging temporal consistency between consecutive frames ($\mathcal{L}_t$), mitigating intra-frame flickering issues at a global (whole frame) or local (block) levels. The final optimization loss function $\mathcal{L}$ is expressed as the weighted sum

$$\mathcal{L}(f, \hat{f}) = \mathcal{L}_r(f, \hat{f}) + \alpha\, \mathcal{L}_h(f) + \beta\, \mathcal{L}_t(f, \hat{f}), \qquad (4)$$

with α and β empirically determined to balance training stability, proximity to the target bitrate, reconstruction quality, and temporal consistency.

1) Reconstruction Loss $\mathcal{L}_r$: The adopted reconstruction loss is simply the *mean squared error* (MSE),

$$\mathcal{L}_r(f, \hat{f}) = \mathrm{MSE}(f, \hat{f}) = \frac{1}{N} \sum_{i=1}^{N} \|f_i - \hat{f}_i\|_2^2. \qquad (5)$$

since this is equivalent to optimizing the PSNR (*peak signal-to-noise ratio*), as $\mathrm{PSNR}(f, \hat{f}) = 10 \log_{10}\big(R^2/\mathrm{MSE}(f, \hat{f})\big)$ (where R is the range of the pixel values), which is the most common metric used in assessing video compressors.

It is possible and would make sense to optimize more sophisticated metrics that correlate better with (human) perceptual quality, such as the well-known MS-SSIM [16]. However, optimizing the PSNR helps to adequately compare the proposed approach with PSNR-optimized conventional codecs.

2) Entropy Loss $\mathcal{L}_h$: The approximate probability models learned during training allow estimating the length of the corresponding optimal codes for the quantized variables. In detail, the length of the optimal codeword for some particular $\hat{z}_i$ is approximately (*i.e.*, ignoring that it has to be integer and assuming that the probability mass function of the quantized

variable is well approximated by the corresponding pdf at that value) equal to $-\log p(\hat{z}_i)$, where $p(\hat{z}_i)$ is approximated by a Gaussian distribution, as described in Section II-A. Similarly, the optimal codeword for some particular $\hat{w}_i$ has length $-\log p(\hat{w}_i|\delta_i)$, as given by (2). Consequently, the total number of bits used by optimal codes for $\bar{z}$ and $\bar{w}$, under these probability models, is approximately given by

$$L(\bar{z},\bar{w}) \simeq \sum_i -\log_2 \mathcal{N}(\bar{z}_i|0,\sigma_i^2) + \sum_j -\log_2 p(\bar{w}_i|\delta_i), \quad (6)$$

where $\mathcal{N}(x|\mu,\sigma^2)$ denotes a Gaussian pdf of mean μ and variance σ^2, computed at x. Finally, since both $\bar{z}$ and $\bar{w}$ are functions of f, we have

$$\mathcal{L}_h(f) = L\Big(q\big(E_\phi(f)\big), q\big(X_\epsilon(E_\phi(f))\big)\Big). \quad (7)$$

By penalizing the entropy of the quantized representations at the bottleneck, rather than simply its dimensionality, we control the amount of information to be used for reconstruction, forcing the network to jointly optimize the reconstruction loss and the bitrate.

3) Temporal Consistency Loss $\mathcal{L}_t$: At low bitrates, the reconstructed video relies heavily on prior information embodied in the reconstruction sub-network D_θ, which may yield inconsistencies between consecutive frames. Since the human visual system is very sensitive to disruptions in the temporal consistency/continuity, this may cause a degradation of the perceptual quality of the reconstructed video sequence. To address this problem, we introduce a short-term temporal consistency loss, $\mathcal{L}_t$, based on the warping error between each pair of subsequent input frames and the corresponding pair of output frames. To calculate the warping we estimate the optical flow between the two input frames, using a *FlowNetC* network [17]. Hence, the temporal loss is then defined as

$$\mathcal{L}_t = \sum_{t=2}^{N} ||M_t \odot (\hat{f}_t - \tilde{f}_{t-1})||_2^2, \quad (8)$$

where $\odot$ denotes pixel-wise product, $\tilde{f}_{t-1}$ is the result of warping the frame $\hat{f}_t$ to time $t-1$ by using the estimated backwards optical flow from f_t to f_{t-1}; and M_t is a binary occlusion mask excluding pixels that are not present in both $\hat{f}_t$ and $\tilde{f}_{t-1}$ (thus are not comparable). M_t is calculated by applying a threshold on the error between the result of warping f_{t-1} with the estimated optical flow and f_t itself.

III. Training and Results

A. Experimental Setup

Our network was trained using PyTorch and optimized for five target bitrates, each using a different parameter setting (see Table I). We use two filter configurations (parameters C1–C3), one for very low bitrates (networks A-C), the other allowing additional information to improve video quality (networks D-E). The training set is composed of 10^4 *high definition* (HD) videos with at least 1080p resolution (1920×1080), randomly chosen from the YouTube-8M dataset [20]. Each video was downscaled to half of its original size in both spatial dimensions (to minimize any existing compression artifacts) and sliced into 32-frame sequences, from which a random 128×128 spatial crop was extracted. The learning rate was fixed at 10^{-4}. Each network was trained until it reached stability on a validation set of 10 videos, also from the YouTube-8M collection, downscaled to 640×360. To avoid overfitting, the training, validation and testing datasets are independent.

TABLE I: Parameters used for the configuration of the five trained models (named A to E) of the proposed video compression scheme.

Network	C1	C2	C3	α	β
A	128	256	128	18	2.5
B	128	256	128	38	3.5
C	128	256	128	59	5.5
D	256	384	256	78	8.5
E	256	384	256	108	11.0

Evaluation was made using 10 uncompressed 1080p videos from the MCL-V database [19], as characterized in Fig. 4. Each evaluation clip was downscaled to a width of 640 pixels, maintaining its aspect ratio. Our solution is evaluated against a *baseline* method, achieved by individually encoding each frame with a DL-based image compression scheme [1]; MPEG-4 Part 2 (libxvid, version 1.3.4); H.264/AVC (libx264, version 142 r2495a, using profile "High Level-2.2"); and H.265/HEVC (libx265, version 1.9, using profile "Main Level-2.1"). Evaluation considers two different quality metrics: the classical PSNR and MS-SSIM [16], which is often used to estimate the perceptual quality of images and videos.

B. Video compression quality assessment

Average rate-distortion curves are presented in Fig. 5(A) and Fig. 5(B) for the RGB color space (for which the network was optimized), including both MS-SSIM and PSNR metrics as a per-frame average. Furthermore, we also show in Fig. 5(C) the results for the Global PSNR in the YUV color space (with 444 sampling), as it represents the commonly used color space and metric for which the comparison codecs are optimized. Although it would improve the results, we performed no special re-training for the YUV color space.

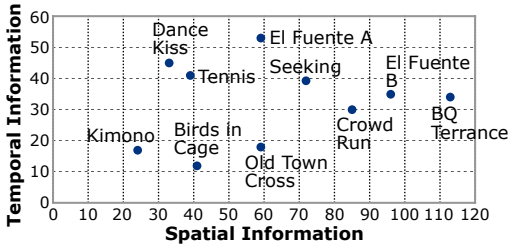

Fig. 4: Characterization of the used test sequences in terms of spatial and temporal information, following the ITU-T Recommendation [18] (as in [19]).

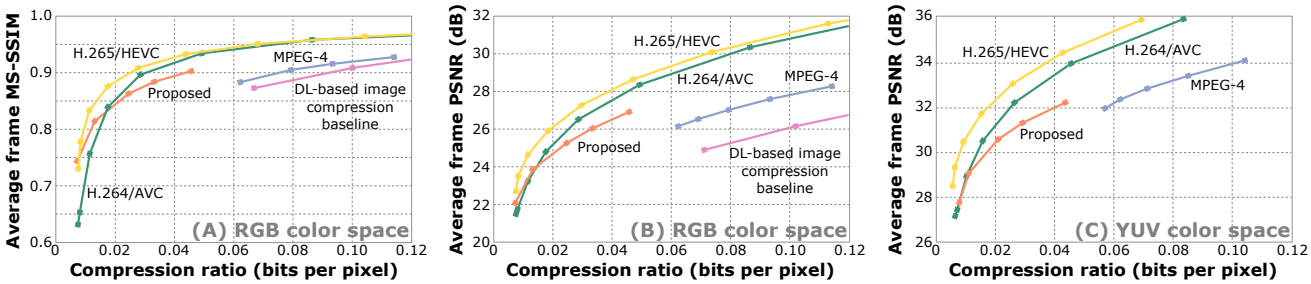

Fig. 5: Summarized quality assessment of video compression schemes for the MCL-V dataset.

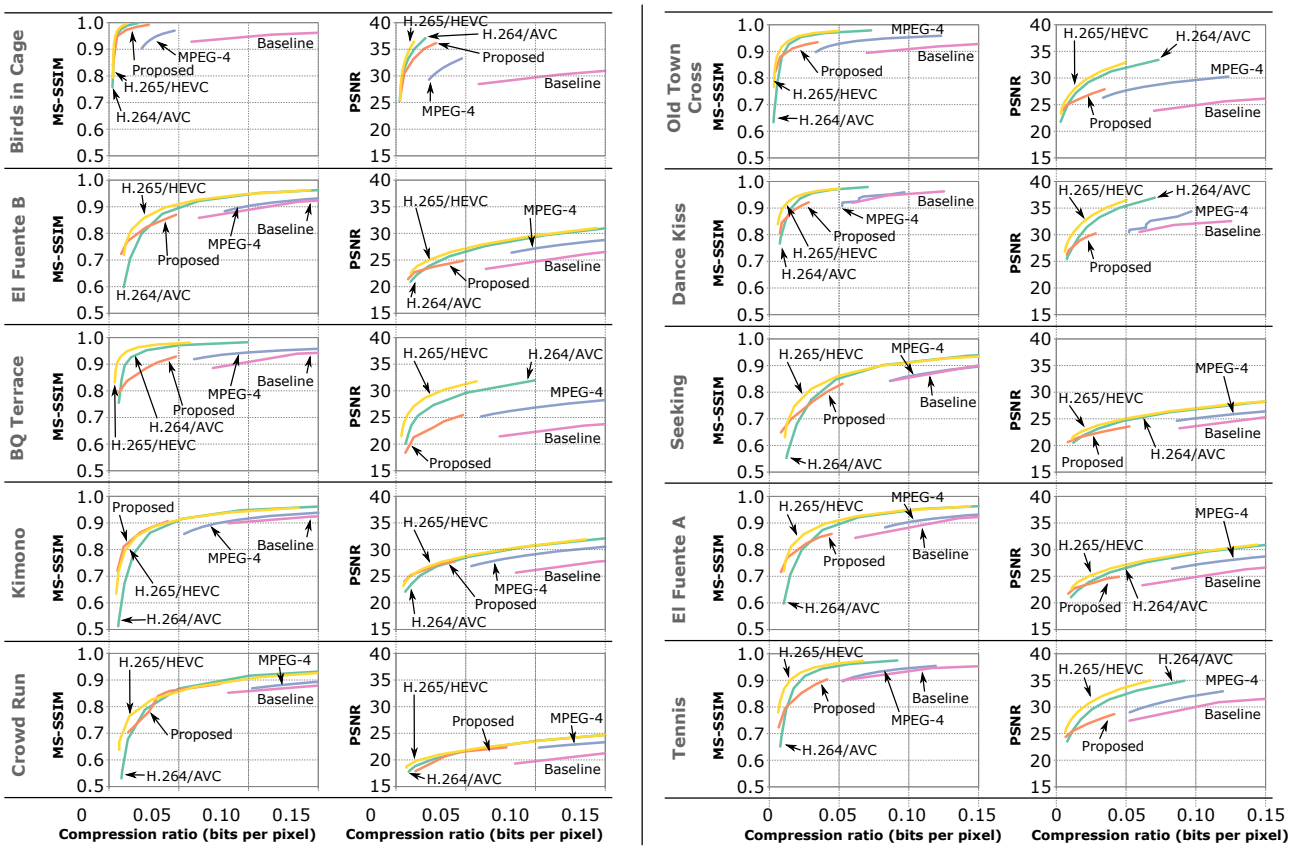

Fig. 6: Individual rate-distortion curves for the compressed videos (RGB color space).

The results show that our video compression network performs better than the baseline and attains similar quality levels compared to MPEG-4 Part2 with significantly lower bitrates. Moreover, it has competitive performance to both H.265/HEVC and H.264/AVC at the lower bitrates, as the network is able to learn a reduced set of features allowing adequate reconstruction of the original video.

To further study the obtained results, Fig. 6 presents individual rate-distortion curves in the RGB color space for each of the considered test sequences. It can be observed that the proposed method tends to attains worse results in video sequences with a high imbalance between spatial and temporal information (e.g., `BQ Terrance`, `Tennis` and `Dance Kiss`), particularly when associated with strong camera motions (e.g., pan, zoom) that are well encoded by motion prediction schemes found in conventional codecs. One particular solution to this problem would be to divide the video sequence into octree-like partitions and allowing different partitions to explore different representations. We leave such an idea for future work. Notwithstanding, even in such videos, the proposed scheme performs acceptable reconstruction of the video, with comparable performance to conventional codecs.

A qualitative assessment of the produced artifacts is also made when using extreme compression ratios. Fig. 7 shows a comparison between crops of frames in `Big Buck Bunny` (extracted from MCL-V [19]), selected because it belongs to a class of videos (synthetic/cartoon) which is almost nonexistent in the training set. Interestingly, the artifacts produced by our network at low bitrates are smoother and may be considered as perceptually more acceptable, compared to the other video

Fig. 7: Example comparison between one video encoded at similar compression ratios, by H.264/AVC, H.265/HEVC and the proposed method. A detailed zoom shows that distortions produced by the proposed approach are more natural.

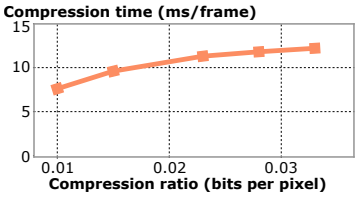

Fig. 8: Average compression time on an Intel Core i7-4770K CPU@3.50GHz and an NVIDIA Titan Xp GPU.

codecs, even at similar PSNR and MS-SSIM values. Such a result follows the research on image compression using deep learning models, where similar conclusions are attained: the artifacts produced by deep learning models are generally more natural than those of heuristic approaches [1], [2], [21].

Finally, Fig. 8 shows the average compression time of the propose scheme, demonstrating that it can operate in real-time.

IV. Conclusion

Video is a complex type of data, for which the existing codecs are extremely well optimized. To be competitive, recently proposed DL-based methods also rely on motion/frame prediction schemes (or equivalent, but similar, methods). In contrast, the video compression problem is herein formulated as a latent space search within a limited entropy constraint. The problem is approached with two transformations: one for projection to the learned latent space, and one for synthesizing the reconstructed video. The bitstream representing the compressed video results from entropy-coding a quantized version of its latent representation. We proposed a rate-distortion optimization framework to train a single spatio-temporal autoencoder for both its reconstruction loss and entropy model. To combat temporal inconsistencies in the decoded videos, the optimization framework was augmented with a short-term temporal loss, encouraging the network to ensure temporal continuity between consecutive frames. The reported results show competitive performances in low bitrates and. Also, the introduced artifacts are visually more pleasing than the unnatural blocking artifacts of standard video codecs.

References

[1] Ballé, Minnen *et al.*, "Variational image compression with a scale hyperprior," in *Int. Conf. on Learning Representations*, 2018.
[2] Johnston, Vincent *et al.*, "Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2018.
[3] Theis, Shi *et al.*, "Lossy Image Compression with Compressive Autoencoders," in *Int. Conf. on Learning Representations*, 2017.
[4] Rippel, Nair *et al.*, "Learned video compression," in *IEEE Int. Conf. on Computer Vision*, 2019.
[5] Park and Kim, "Deep predictive video compression with bi-directional prediction," *arXiv preprint arXiv:1904.02909*, 2019.
[6] Liu, Chen *et al.*, "Neural video compression using spatio-temporal priors," *arXiv preprint arXiv:1902.07383*, 2019.
[7] Lu, Ouyang *et al.*, "DVC: An end-to-end deep video compression framework," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2019.
[8] Wu, Singhal, and Krähenbühl, "Video compression through image interpolation," in *European Conf. on Computer Vision*, 2018.
[9] Djelouah, Campos *et al.*, "Neural inter-frame compression for video coding," in *IEEE Int. Conf. on Computer Vision*, 2019.
[10] Chen, He *et al.*, "Learning for video compression," *IEEE Trans. on Circuits and Systems for Video Technology*, 2019.
[11] Chen, Liu *et al.*, "Deepcoder: A deep neural network based video compression," in *IEEE Visual Communications and Image Processing*, 2017.
[12] Habibian, Rozendaal *et al.*, "Video compression with rate-distortion autoencoders," in *IEEE Int. Conf. on Computer Vision*, 2019.
[13] Lombardo, HAN *et al.*, "Deep generative video compression," in *Advances in Neural Information Processing Systems*, 2019.
[14] Sullivan, Ohm *et al.*, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012.
[15] Gersho and Gray, *Vector Quantization and Signal Compression*. Springer, 1992.
[16] Wang, Simoncelli, and Bovik, "Multiscale structural similarity for image quality assessment," in *Asilomar Conf. on Signals, Systems Computers*, 2003.
[17] Ilg, Mayer *et al.*, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2017.
[18] ITU-T RECOMMENDATION, "Subjective video quality assessment methods for multimedia applications," *Int. telecommunication union*, 1999.
[19] Lin, Song *et al.*, "MCL-V: A streaming video quality assessment database," *J. of Visual Communication and Image Representation*, vol. 30, pp. 1–9, 2015.
[20] Abu-El-Haija, Kothari *et al.*, "Youtube-8m: A large-scale video classification benchmark," in *ArXiv e-prints*, 2016. [Online]. Available: http://arxiv.org/abs/1609.08675
[21] Ballé, Laparra, and Simoncelli, "End-to-end optimized image compression," in *Int. Conf. on Learning Representations*, 2017.