Hierarchical Temporal Structure for End-to-End Neural Network-based Video Compression

Jean Bégaint*, Fabien Racapé*, Simon Feltman, Akshay Pushparaja
InterDigital, AI Lab
4410 El Camino Real, Suite 120, Los Altos, CA, 94022
firstname.lastname@interdigital.com

Abstract

This paper presents an end-to-end Artificial Neural Network (ANN)-based compression framework in response to the video compression task of the Challenge for Learned Image Compression (CLIC) at CVPR 2021. In this framework, the video frames are divided into Groups Of Pictures (GOPs) in which each frame can be encoded in Intra or Inter mode. In Intra mode, an auto-encoder compresses the pixel values directly. For Inter frames, we leverage bi-directional prediction with reference frame signalling, allowing for efficient hierarchical GOP temporal structures. The motion information, computed using the luminance, and prediction residuals are compressed using dedicated auto-encoder structures, in which the layers are conditioned based on the GOP structure. The network is trained fully end-to-end, from scratch. The results demonstrate the promises of end-to-end approaches.

1. Introduction

Artificial Neural Networks (ANN) have successfully replaced handcrafted descriptors and classical algorithms for a large number of computer vision applications. In the domain of image compression, recent state-of-the-art ANN-based methods [10] show promising performances, as they now compare with the best performing traditional codecs, such as the video compression standard H.266/VVC [3] in still picture mode. Digital video compression has seen a huge industrial effort over the last 30 years, leveraging competition in standardization activities to optimize inter-operable standards. State-of-the-art standards such as H.265/HEVC [12], H.266/VVC [3] or AV1 [5] are the result of years of research and iterations over the successful hybrid coding approach. In this context, images and video frames are partitioned into non-overlapping rectangular blocks of different sizes and shapes. Temporal and spatial redundancies are reduced by performing Inter frame and Intra frame predictions, respectively. Then, spatial frequencies of the prediction residuals are decorrelated using linear transforms. The obtained coefficients are quantized and entropy coded alongside the necessary metadata which signals all the decisions made by the encoder, such as prediction modes, partitioning, frame inter-dependencies, etc.

Even though ANN-based methods now seem ubiquitous in computer vision, their usage in image and video compression is fairly recent, but has shown a promising curve of performance improvement over the past couple of years. Neural networks can complement or replace parts of the existing hybrid video coding approach or be trained end-to-end to directly encode and decode images or videos. In the first approach, one or several modules of traditional codecs can be replaced by neural network-based methods, such as post filters, intra or inter predictions, or transforms. For instance, the Joint Video Exploration Team, which designed H.266/VVC is experimenting with ANN-based loop filters that could replace or enhance existing deblocking filters, Sample Adaptive Offsets or Adaptive Loop Filters.

The end-to-end approach consists in designing a neural network based video compression codec, disrupting the hybrid model of traditional codecs, which remains highly challenging. In this paper, we present a framework taking on this second approach in which a series of ANN-based autoencoders are trained end-to-end to efficiently compress videos. In particular, the proposed method implements a conditional bi-directional prediction scheme, leveraging well-studied hierarchical temporal structures to reduce inter-frame redundancies. Several works have been previously published regarding neural networks targeting video compression [6, 9, 1]. These approaches consider a temporal structure referred to as a Low Delay configuration, where directly neighboring causal frames are considered for inter prediction. Like in [14], our proposal explores bi-directional prediction and hierarchical GOP (Group Of Picture) structures, while considering a different model structure from the recurrent approach in [14].

*Equal contribution

The remainder of the paper is organized as follows. In Section 2, we describe the proposed methods and temporal structure for efficient compression. Then, we describe the experimental setup, including implementation and training details in Section 3, before discussing the corresponding tests and results in Section 4.

2. Proposed Architecture

The proposed network is composed of 3 parts, all autoencoders, an intra module to encode the keyframes, an inter module to jointly estimate and compress the motion information, and a residual module to learn and encode efficiently a compensation for the prediction error.

2.1. Intra and Inter coding of the video frames

The architecture described in the paper is based on the low-delay network proposed in [1], extended to efficiently perform bi-directional prediction, and to directly support the YUV 4:2:0 frame format.

The image and residual autoencoders structure is inspired by the proposal described in [8], that is able to handle YUV 4:2:0 format, where the surface of the chrominance channels is a fourth of the surface of the luminance. Figure 1 shows the decoder process in the case of an Intra coded frame. The bitstream is first parsed and the latent tensor of M channels is entropy decoded. Then, a series of transpose convolutions with ReLU activations and N channels are applied. When a tensor has the spatial dimension of the chroma components, *i.e.* $N \times h/2 \times w/2$, a 1×1 convolution with $2N$ output channels is applied. The output tensor is then split into 2 parts of N channels each so that different final layers can be applied to luminance and chrominance. The last layer of the luminance channel applies a transpose convolution with a stride of 2 and an output of 1 channel, whereas no stride is needed to output the 2 channels of chrominance. As in other auto-encoder solutions, the encoder symmetrically produces and entropy encodes a latent representation derived from the input content using a fully convolutional approach.

In inter mode, the frames can be predicted from previously decoded frames. The elements to encode for each frame consist of a motion flow, residual and metadata to signal which reference frames are used. The residual corresponds to the prediction error, it has the same shape as the input image and can be encoded and decoded the same way as for the Intra mode described above. The motion flow corresponds to a map of pixel displacements which is used to warp the reference frames onto the current picture to predict. In the next section, we detail how these elements are computed and compressed.

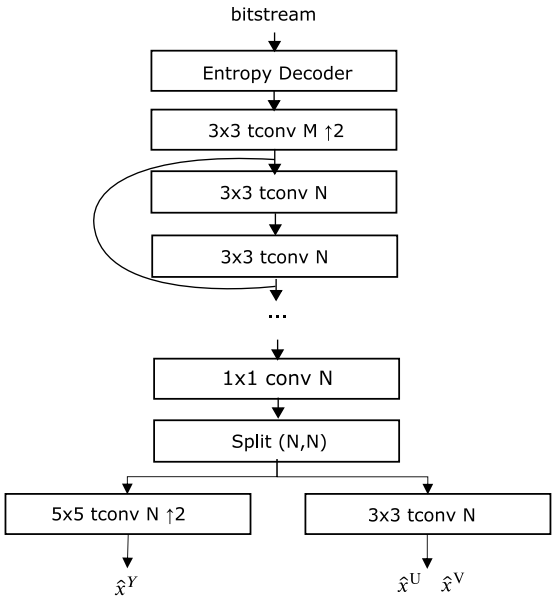

Figure 1. Example decoder architecture for YUV 4:2:0 output format (intra and residual).

2.2. Estimating and encoding the motion from the luminance channel

We propose to compute the motion information using the luminance component only. The luminance and the chrominance of the reference image are then warped onto the current frame separately to form the prediction. When representing frames in the YUV 4:2:0 format, most of the information is contained in the luma component and the chroma channels are effectively of low entropy. Besides simplifying the network structure, this allows the network to learn from only relevant inputs.

The luminance component of each frame and its references (x^Y,$\hat{x}^Y_{ref}(i)$) are used to compute the motion. They are first concatenated to produce a tensor of size $2 \times h \times w$ where h and w denote the height and width of the frames, respectively. The motion, *i.e.*, the horizontal and vertical components of the displacement of each sample, is then computed, which then also corresponds to a $2 \times h \times w$ tensor. Optionally, a component can be added, called scale field [1] which aims at introducing blur when flow-based prediction is not good enough, *e.g.*, when occlusions occur or objects move out of bounds. This motion tensor is encoded using a convolutional autoencoder with an entropy bottleneck. The output of this stage consists of a tensor containing the motion vector components and an optional field scale. We use the $\tanh$ activation function to predict values in the $[-1, 1]$ range, relative to the frame size.

The reconstructed reference frame is warped with these tensors to produce the predicted image x^{YUV}_{pred}. During the warping, the range $[-1, 1]$ of the motion corresponds to motion vectors with the ranges $[-width, width]$, respectively

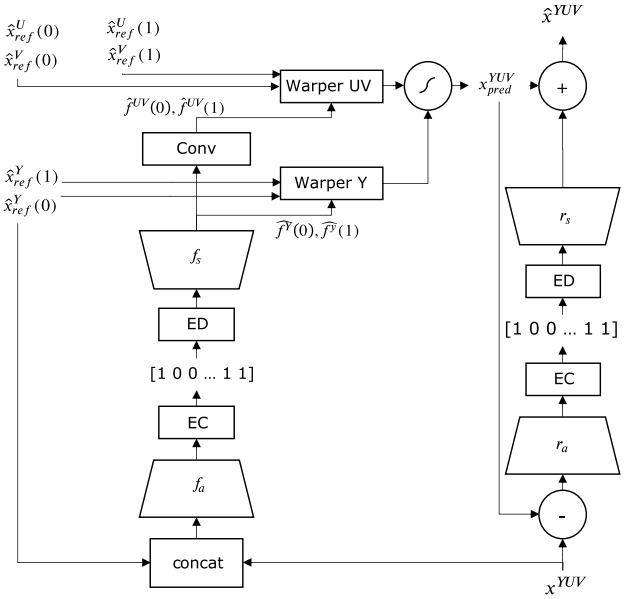

Figure 2. Coding structure for inter coded frames. The luminance component of the current frame as well as the reference pictures is used to derive the compressed motion information. At the decoder end, a strided convolution downsamples the motion flow to the resolution of chrominance before warping

$[-height, height]$ for the horizontal, respectively vertical, component of each sample's displacement. The residual r^{YUV} is obtained by subtracting the predicted values from the current source image at the encoder.

$$r^{YUV} = x^{YUV} - x^{YUV}_{pred} \tag{1}$$

Similar to the image and motion information, the residual is encoded using an autoencoder architecture. This process is also lossy, which means that the reconstructed residual $\hat{r}^{YUV}$ is not equal to the previously constructed residual r^{YUV}. For each inter-predicted image, the bitstream thus contains two parts: one for the motion information and one for the residuals of the prediction.

The reconstructed frame finally corresponds to:

$$\hat{x}^{YUV} = x^{YUV}_{pred} + r^{YUV} \tag{2}$$

In the case of YUV 4:2:0 format, the size of U and V component is generally $width/2$ and $height/2$. The absolute displacements of each chrominance sample then corresponds to half of those of the Y component. A downsampling operation is necessary to derive motion information at the correct spatial size for the chrominance, as shown in Figure 2. This down sampling operation consists of a 3×3 convolution with a stride of 2. The weights are learned during training, which helps catching the phase between the luminance and chrominance displacements.

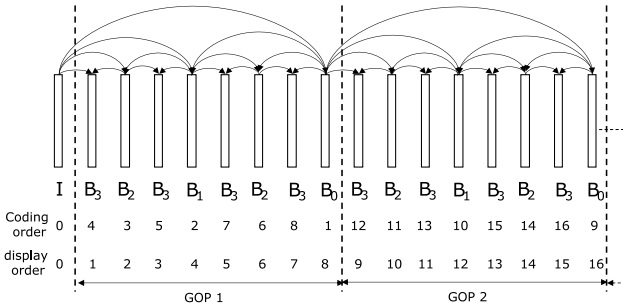

Figure 3. Prediction inter-dependencies in a sequence of two hierarchical GOP of 8 frames.

2.3. Bi-directional prediction and hierarchical GOPs

The Random Access configuration is an efficient temporal prediction structure relying on repeating Groups of Pictures (GOPs), which consist of the minimal temporal time frame structure. Specific instant random access points, consisting of a purely Intra coded frames, enable the decoder to start the decoding since no temporal context is required to initialize the parsing. Figure 3 illustrates such a structure in the case of a GOP of 8 frames. The first frame is an Intra frame, or I-frame, meaning that it does not depend on other frames to be decoded. It can then be used as a random-access point, where a decoder can start decoding a sequence. In broadcast, they are typically separated by a second of video, which enables TV viewers to switch channels and start decoding the new channel they selected, and not wait for too long for the video to start being displayed. However, these frames usually cost a lot of bits to transmit since they are not predicted using previously decoded content. Between I-frames, the remaining frames are predicted using the previously decoded frames. In the structure of Figure 3, one can notice that the coding order is different from the order of display. This enables the encoder to predict the frames using past and future previously reconstructed pictures, allowing to bypass occlusions and appearing/disappearing objects. These frames are hence called B frames for bi-directional prediction.

The structure follows a hierarchical pattern with frames of type B_0, B_1, B_2, and B_3. The B_0 of each GOP is the first frame to be coded, it is predicted using the last key frame (I or B_0) from previous GOPs, e.g., frame 8 in display order is predicted from frame 0. The following frames in coding order can be predicted using past and future frames, as depicted by the arrows. Frames B_1 can use frames of type I, B_0, frames B_2 can be predicted from frames I, B_0 and B_1 etc. The distance between the current frame and the reference frame then varies depending on the decisions made by the encoder. We use conditional activation to condition the motion and residual encoders/decoders to the level of the current B frame. This allows the network to learn more ef-

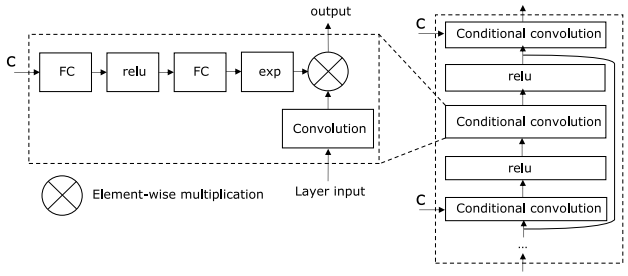

Figure 4. Convolutions / transpose convolutions with conditional activation in the autoencoders for residual and motion information.

ficient representations, as the lower hierarchical levels are usually subject to larger motions, while the quality of the reference frames decreases with the GOP depth. We use a one-hot encoding of the 4 possible frame levels (B0 to B3), we also experimented with directly encoding the frame positions in the GOP (1 to 8) but it did not performed as well. Figure 4 shows such conditional convolutions, where "FC" stands for Fully Connected and exp is the exponential activation function.

3. Experimental setup

Architecture The network is implemented in PyTorch with the help of the CompressAI library [4]. The main buildings blocks are hyperprior networks from [2] for the keyframe, inter and motion autoencoders. We use ReLU activations instead of GDN, replace each convolutions by two consecutive residual blocks. Uniform quantization and rounding are mixed to relax the non-differentiable quantization operations during training [11]. The entropy coding is performed using an ANS (asymmetric numeral systems) coder, a faster variant of arithmetic coding [7].

Training For the training data, we randomly extracted and cropped consecutive frames from the CLIC2021 dataset. We used approximately 200k samples. Frames are grouped in temporal chunks of *gop_size* elements. To accelerate the training we don't propagate the gradients of the reference frames used for inter prediction (*detach* in PyTorch, *stop_gradients* in Tensorflow).

We trained for 150 epochs using the Adam optimizer with a starting learning rate of $1e^{-4}$, a batch size of 8 and cropped frames of size 256×256 pixels. We decay the learning rate by 0.1 whenever the loss reaches a plateau. The network is first optimized for MSE to improve convergence and stability, we then switch to the MS-SSIM metric at 80% of the total training steps, and also increase the patch size to reduce border artifacts. Training is performed on a single RTX 8000 GPU.

Quantized scale transform To ensure a correct decoding of the latents across devices and platforms, some floating point operations need to be replaced with integer arithmetic. More specifically, the decoded scales from the hyperprior transforms need to be deterministic for the latents to be decoded properly by the entropy coder. To do so, we first train the network in floating point and then perform post-training quantization (sometimes also called *static quantization*) and calibration of the scale transforms. No performance impact on the actual bitstream was measured after quantization.

4. Results

Test conditions The proposed framework has been evaluated in the conditions of the video compression challenge for CLIC 2021 (Challenge on Learned Image Compression) CVPR workshop. The participants are provided with a large training set of hundreds of videos from which a test set of 100 clips of 2 seconds at 30Hz is selected. This corresponds to 60 frames per clip. Sequence resolutions vary between 1440×720, 1280×720, 960×720 and 959×720. The sequences are in the YUV 4:2:0 color format, *i.e* the chrominance component (U and V) are 2 times smaller in both vertical and horizontal dimension than the luminance channel (Y).

The leader-board of the challenge ranks the responses using the MS-SSIM metric [13] measure on the decoded sequences. Only one operational bit-rate point of approximately 1Mb/s is considered. The total size of the submission, *i.e.* the sum of all the encoded sequence binaries and the decoder executable, is limited as follows:

$$S_{bitstreams} + 0.019 * S_{decoder} < 1,309,062,500 \quad (3)$$

This penalizes the decoder model size and prevents proposals from overfitting on the whole training set from which the test set is extracted. The proposed decoder is approximately 44MB, which leaves around 24MB to encode the 100 sequences, or approximately 0.0342 bpp.

Objective results Objectively, our network achieves MS-SSIM/PSNR as 0.9520/29.77dB on the validation set. The total size of the decoder in 44MB, and 720p frames are decoded in 0.2s on average (non-optimized code running on GPU).

5. Conclusion

In this paper, we presented a end-to-end video compression framework which leverages efficient hierarchical temporal structure for interframe predictions. Our model is able to directly encode YUV 4:2:0 frames, rely on motion and residual estimators learned from scratch, and conditional convolutions to handle the disparity of motions between GOP levels. The results show the promises of an end-to-end bidirectional approaches for ANN-based video

compression. Several future directions can be considered: generalizing the model to more complex motion modeling, better handling of screen-content videos, scene cuts, and generalizing to a more flexible inter prediction structure.

References

[1] Eirikur Agustsson, David Minnen, Nick Johnston, Johannes Balle, Sung Jin Hwang, and George Toderici. Scale-space flow for end-to-end optimized video compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8503–8512, 2020. 1, 2

[2] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. *arXiv preprint arXiv:1802.01436*, 2018. 4

[3] B. Bross, J. Chen, J. R. Ohm, G. J. Sullivan, and Y. K. Wang. Developments in international video coding standardization after avc, with an overview of versatile video coding (vvc). *Proceedings of the IEEE*, pages 1–31, 2021. 1

[4] Jean Bégaint, Fabien Racapé, Simon Feltman, and Akshay Pushparaja. Compressai: a pytorch library and evaluation platform for end-to-end compression research, 2020. arXiv: 2011.03029. 4

[5] Y. Chen, D. Murherjee, J. Han, A. Grange, Y. Xu, Z. Liu, S. Parker, C. Chen, H. Su, U. Joshi, C. Chiang, Y. Wang, P. Wilkins, J. Bankoski, L. Trudeau, N. Egge, J. Valin, T. Davies, S. Midtskogen, A. Norkin, and P. de Rivaz. An overview of core coding tools in the av1 video codec. In *2018 Picture Coding Symposium (PCS)*, pages 41–45, 2018. 1

[6] Abdelaziz Djelouah, Joaquim Campos, Simone Schaub-Meyer, and Christopher Schroers. Neural inter-frame compression for video coding. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6421–6429, 2019. 1

[7] Jarek Duda. Asymmetric numeral systems. *arXiv preprint arXiv:0902.0271*, 2009. 4

[8] Hilmi E. Egilmez, Ankitesh K. Singh, Muhammed Coban, Marta Karczewicz, Yinhao Zhu, Yang Yang, Amir Said, and Taco S. Cohen. Transform Network Architectures for Deep Learning based End-to-End Image/Video Coding in Subsampled Color Spaces. *arXiv:2103.01760 [cs, eess]*, Feb. 2021. arXiv: 2103.01760. 2

[9] Amirhossein Habibian, Ties Van Rozendaal, Jakub Tomczak, and Taco Cohen. Video Compression With Rate-Distortion Autoencoders. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7032–7041, Seoul, Korea (South), 2019. IEEE. 1

[10] David Minnen, Johannes Ballé, and George D Toderici. Joint autoregressive and hierarchical priors for learned image compression. In *Advances in Neural Information Processing Systems*, pages 10771–10780, 2018. 1

[11] David Minnen and Saurabh Singh. Channel-wise autoregressive entropy models for learned image compression. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 3339–3343. IEEE, 2020. 4

[12] Gary J Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the high efficiency video coding (hevc) standard. *IEEE Transactions on circuits and systems for video technology*, 22(12):1649–1668, 2012. 1

[13] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pages 1398–1402. Ieee, 2003. 4

[14] Ren Yang, Fabian Mentzer, Luc Van Gool, and Radu Timofte. Learning for Video Compression with Hierarchical Quality and Recurrent Enhancement. *arXiv:2003.01966 [cs, eess]*, Aug. 2020. arXiv: 2003.01966. 1