A Video Compression Framework Using an Overfitted Restoration Neural Network

Gang He[1,3], Chang Wu[1], Lei Li[1], Jinjia Zhou[2], Xianglin Wang[3], Yunfei Zheng[3], Bing Yu[3], Weiying Xie[1]

[1] Xi'dian University, Xi'an, China
[2] Graduate School of Science and Engineering, Hosei Univeristy, Tokyo, Japan
[3] Beijing Kuaishou Technology

Abstract

Many existing deep learning based video compression approaches apply deep neural networks (DNNs) to enhance the decoded video by learning the mapping between decoded video and raw video (ground truth). The big challenge is to train one well-fitted model (one mapping) for various video sequences. Different with the other applications such as image enhancement whose ground truth can only be obtained in the training process, the video encoder can always get the ground truth which is the raw video. It means we can train the model together with video compression and use one model for each sequence or even for each frame. The main idea of our approach is building a video compression framework (VCOR) using overfitted restoration neural network (ORNN). A lightweight ORNN is trained for a group of consecutive frames, so that it is overfitted to this group and achieves a strong restoration ability. After that, parameters of ORNN are transmitted to the decoder as a part of the encoded bitstream. At the decoder side, ORNN can perform the same strong restoration operation to the reconstructed frames. We participate in the CLIC2020 challenge on P-frame track as the team "WestWorld".

1. Introduction

Recently, the deep neural networks (DNNs) based video compression approaches have developed and aroused widespread interest [5–7, 12, 18–20]. For example, the embedded hand-crafted modules are replaced with DNNs in the works [5, 7, 12, 18], such as intra prediction, motion estimation, and in-loop filter. An end-to-end deep video compression framework has been developed in [6]. The existing works considered a video compression problem being similar to other image processing problems, e.g., super resolution, denoising and deblurring, where in the real-world use case (in the testing process), the right answer (ground truth) cannot be obtained. During the training process, huge data is fed to DNN to make the output close to the ground truth (original video or the desired prediction) and make sure not overfitted to the training dataset. It is a big challenge to design a well-structured and well-trained DNN.

In this paper, we consider the unique characteristic of a video compression, where in the real-world use case, ground truth is the input raw video and can be obtained in an encoder. We propose a video compression framework named VCOR which combining a conventional video codec with an overfitted restoration neural network (ORNN). The ORNN is used to restore the conventionally compressed video in an overfitted way. At the encoder side, ORNN is trained online with the reconstructed frames from the conventional encoder as the input and the raw frames as the ground truth. Overfitting for a group of consecutive frames is activated on purpose to achieve a strong restoration ability. To transfer this ability from the encoder to the decoder, the transmitted information contains not only the syntax of a conventional codec (also called bitstream), but also the overfitted parameters of ORNN. At the decoder side, with the received parameters, an inference operation of ORNN is performed on the decoded frames to implement the strong restoration. Moreover, to achieve the performance gain by utilizing the overfitting, a lightweight ORNN is designed delicately to restore the video better, while using fewer additional transmitted parameters.

2. The proposed video compression framework

2.1. Overview of VCOR

As shown in Fig. 1, the proposed video compression framework VCOR contains the conventional compression flow and additional restoration flow, implemented by a con-

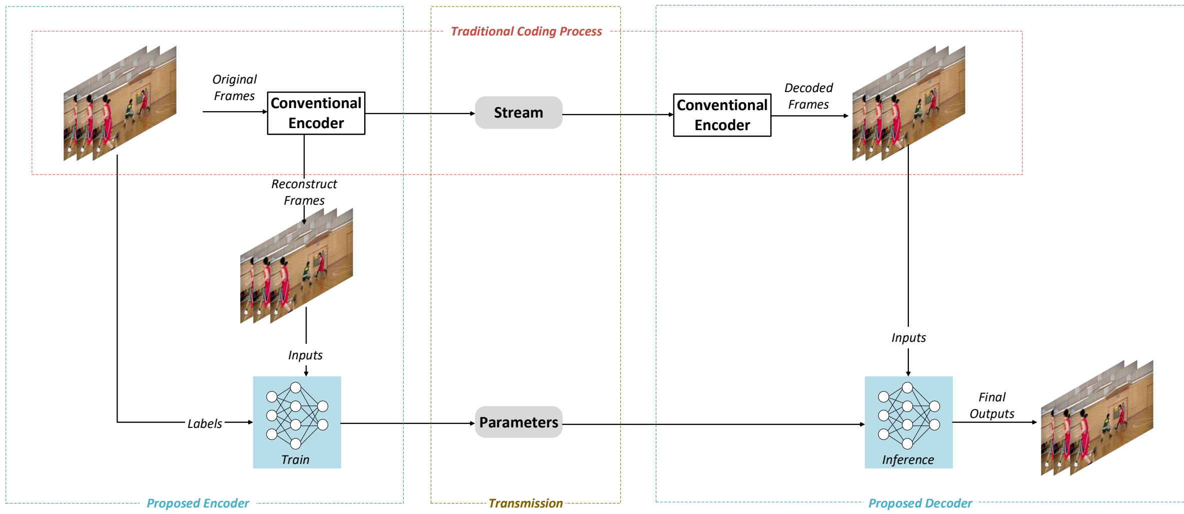

Figure 1. The proposed video compression framework VCOR contains the conventional compression and additional restoration. In an encoder, ORNN is trained online with the reconstructed frames from a conventional encoder as the input and the raw frames as the ground truth. The overfitting is realized by training it only with a group of consecutive frames. The overfitted parameters of ORNN are transmitted as a part of an encoded bitstream. In a decoder, with the transmitted parameters, an inference operation of ORNN is performed on the decoded frames to generate the final result.

ventional codec and ORNN respectively. The procedure is explained as follows. The input video can be divided into the groups of consecutive raw frames and compressed group by group. In the VCOR encoder, a group of consecutive raw frames are encoded first by a conventional encoder, which outputs the bitstream for transmission and reconstructed frames for ORNN. Then, ORNN is trained online with reconstructed frames as the input and raw frames as the label. The overfitting can be realized by training it only with a small dataset that is a group of consecutive frames. The overfitted parameters are transmitted from an encoder to a decoder, being considered as the information which function similarly to the hand-crafted bitstream of a conventional compression. It needs to be noted that the overfitted parameters only benefit the group which has been trained, and need to be transmitted for each group of a video. In the VCOR decoder, the bitstream is first decoded by a conventional decoder. Then, with the received parameters learned in the VCOR encoder, ORNN performs an inference operation on the decoded frames for the final result. The same strong restoration can be done as in the VCOR encoder.

2.2. Overfitted Restoration Neural Network

In VCOR, the performance gain depends on not only the restored quality, but also the additionally transmitted information. The number of parameters that affects both of them becomes critical for the design of the neural network. "The deeper, the better" is not suitable for our case. It is because although the deeper neural network can restore the video better, while times more parameters have to be transmitted. Therefore, in the proposed framework, a delicate lightweight DNN needs to be designed, which tries to achieve a better tradeoff between the quality restoration and number of parameters. Another advantage resulting from a lightweight DNN is the faster processing speed compared with other post-porcessing methods.

Network architecture. The overall architecture of the proposed network, which called ORNN, is illustrated in Fig. 2. Which take reconstructed target frame $\hat{I}_t$ and reference frame as input which is denoted as HR and original target frame I_t as label. ORNN is aimed to generate a high quality frame O_t, which is close to the ground truth frame I_t. According to our design ideas, it can be divided into three parts, including a base branch, two multi-scale branches, and a channel attention reconstruction module. Three parts are described as follows.

(1) Base branch. We first design a very simple structure called base branch that illustrated at the bottom of Fig. 2. Stacked reconstructed frame and it's reference frame HR are directly send into the base branch, then different levels of hidden feature maps are output to reconstructed module. The whole process is defined as:

$$h_0^0, \ldots, h_0^n = f_B(HR) \tag{1}$$

which $h_0^0, \ldots, h_0^n$ representing different levels of hidden feature maps. f_B is our base branch network. It need to mentioned that in most of multi-frame restoration works [1, 3, 10, 17, 20], neighbor frames always be explicitly aligned by motion estimation module before sent to feature extraction module. It dose improve the visual quality, while a large number of parameters must be used for this operation. It's not economic for our system.

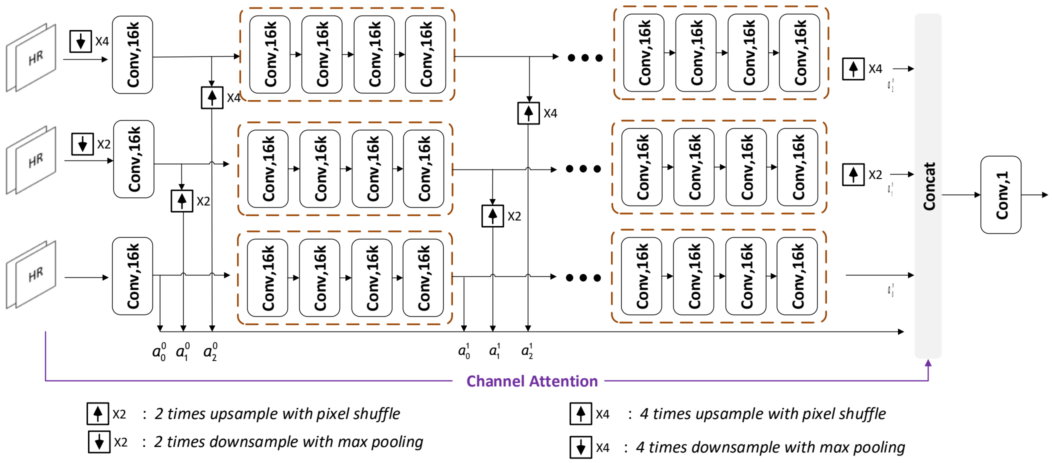

Figure 2. Structure of our overfitting neural network. It can be simply devided into three branch form the bottom to the top, namely base branch, multi-scale branch 1, multi-scale branch 2. Three branches share the same parameters and are used to extract different scale information. Channel attention is intorduced to learn the importance of concatenated features.

(2) Multi-scale branch. As mentioned above, we send reconstructed frames HR directly to base branch network without any alignment operation. At the same time, due to the limitation of parameters, we are not allowed to design a very deep network. Thus our base branch has only extremely limited receptive field, which is hard to extract information form those neighbor frames with large shift, especially those import edge information. In recent years, there are many methods can be used to expand receptive fields without increase the deep of the network, such as deformable convnet [2,22], non-local convnet [14]. But those methods also bring either times more parameters or huge amount of calculation. Inspired by works [9,13], we choose multi-scale strategy for broader receptive field while only increase less than half the amount of calculation. Sharing network weights across scales to significantly save parameters and speed up training. To preserve edge information, we choose maxpooling for downsampling as

$$\begin{aligned} A_1 &= MaxPooingX2(HR) \\ A_2 &= MaxPooingX4(HR) \end{aligned} \tag{2}$$

where maxpooling is used for 2 times and 4 times downsampling. A_1 and A_2 are respectively send to two multi-scale branch as

$$\begin{aligned} h_1^0, \ldots, h_1^n &= f_{MS1}(A_1) \\ h_2^0, \ldots, h_2^n &= f_{MS2}(A_2) \end{aligned} \tag{3}$$

where f_{MS1} and f_{MS2} is our multi-scale branch network that share the same weights with the base branch. $h_1^0, \ldots, h_1^n$ and $h_2^0, \ldots, h_2^n$ are outputs of multi-scale branch, which will concatenate with $h_0^0, \ldots, h_0^n$ and then send to the channel attention reconstruction module.

(3) Channel attention reconstruction module. In our work, lots of hidden features that come from three branches are sent to reconstruction module, but generally these hidden features are not of equal importance thus attention mechanism is introduced as a guidance to bias the allocation of available features towards the most informative components. Hidden features from different branch have different scale, which need to upsample before attention

$$\begin{aligned} a_0^0, \ldots, a_0^n &= h_0^0, \ldots, h_0^n \\ a_1^0, \ldots, a_1^n &= f_{up2}(h_1^0), \ldots, f_{up2}(h_1^n) \\ a_2^0, \ldots, a_2^n &= f_{up4}(h_2^0), \ldots, f_{up4}(h_2^n) \end{aligned} \tag{4}$$

where f_{up2} and f_{up4} are separately $\times 2$ and $\times 4$ upsampling function with pixel shuffle [11] . After upsample, all features with the same scale are concatenated then perform channel attention

$$\begin{aligned} H_{all} &= [a_0^0, \ldots, a_0^n, a_1^0, \ldots, a_1^n, a_2^0, \ldots, a_2^n] \\ F_{Att} &= f_{Att}(HR) \otimes H_{all} \end{aligned} \tag{5}$$

where $[,]$ and $\otimes$ separately denote concatenation and element-wise product. f_{Att} is channel attention function using neural network mainly referenced methods [16, 21]. Feature F_{Att} is used for final reconstruction.

$$O_t = f_{Rec}(F_{Att}) \tag{6}$$

f_{Rec} is our reconstruction function with a simple one layer convolutional neural network.

The hyper-parameter n and k directly influence the depth and width of our model which is influenced by specific video and can be adjust adaptively. Here we simply set it to 6 and 2.

3. Experiments

Training Details. During the training, the frames are cropped into 256x256 non-overlapping patches. The batch

	Float32	Float16
Model size	0.856 MB	0.428 MB

Table 1. Model size.

size is set as 16. The learning rate is set as 0.001 in first two epoch and then divided by a factor of 10 after 40 epoch. The Adam optimizer [4] is used by setting $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The proposed ORNN is implemented with Pytorch 1.2.0 [8]. Our model contains 224.4k parameters totally. To ensure a precise backpropagation, parameters adopt 32-bit floating point format during the training, while are converted to the ones with 16-bit floating point format after the training to reduce the transmitted information. Table 1 shows the different model size between 32-bit floating point format and 16-bit floating point format. In CLIC competition, we use one model fit all test data.

Loss Function. We choose MS-SSIM [15] as our loss function. Which can be described as follow.

$$\begin{aligned}
&l(\mathbf{x}, \mathbf{y}) = \tfrac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \\
&c(\mathbf{x}, \mathbf{y}) = \tfrac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \\
&s(\mathbf{x}, \mathbf{y}) = \tfrac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \\
&\text{MS-SSIM}(\mathbf{x}, \mathbf{y}) = [l_M]^{\alpha_M} \cdot \textstyle\prod_{j=1}^{M} [c_j]^{\beta_j} \, [s_j]^{\gamma_j}
\end{aligned} \tag{7}$$

References

[1] Jose Caballero, Christian Ledig, Andrew Aitken, Alejandro Acosta, Johannes Totz, Zehan Wang, and Wenzhe Shi. Real-time video super-resolution with spatio-temporal networks and motion compensation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4778–4787, 2017.

[2] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017.

[3] Muhammad Haris, Gregory Shakhnarovich, and Norimichi Ukita. Deep back-projection networks for super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1664–1673, 2018.

[4] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[5] Tianyi Li, Mai Xu, Ren Yang, and Xiaoming Tao. A densenet based approach for multi-frame in-loop filter in hevc. In *2019 Data Compression Conference (DCC)*, pages 270–279. IEEE, 2019.

[6] Guo Lu, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Chunlei Cai, and Zhiyong Gao. Dvc: An end-to-end deep video compression framework. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11006–11015, 2019.

[7] Woon-Sung Park and Munchurl Kim. Cnn-based in-loop filtering for coding efficiency improvement. In *2016 IEEE 12th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP)*, pages 1–5. IEEE, 2016.

[8] Adam Paszke, Sam Gross, Soumith Chintala, and Gregory Chanan. Pytorch. *Computer software. Vers. 0.3*, 1, 2017.

[9] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[10] Mehdi SM Sajjadi, Raviteja Vemulapalli, and Matthew Brown. Frame-recurrent video super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6626–6634, 2018.

[11] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1874–1883, 2016.

[12] Rui Song, Dong Liu, Houqiang Li, and Feng Wu. Neural network-based arithmetic coding of intra prediction modes in hevc. In *2017 IEEE Visual Communications and Image Processing (VCIP)*, pages 1–4. IEEE, 2017.

[13] Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiaya Jia. Scale-recurrent network for deep image deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8174–8182, 2018.

[14] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7794–7803, 2018.

[15] Zhou Wang and Qiang Li. Information content weighting for perceptual image quality assessment. *IEEE Transactions on image processing*, 20(5):1185–1198, 2010.

[16] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–19, 2018.

[17] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *International Journal of Computer Vision*, 127(8):1106–1125, 2019.

[18] Ning Yan, Dong Liu, Houqiang Li, and Feng Wu. A convolutional neural network approach for half-pel interpolation in video coding. In *2017 IEEE international symposium on circuits and systems (ISCAS)*, pages 1–4. IEEE, 2017.

[19] Ren Yang, Mai Xu, Tie Liu, Zulin Wang, and Zhenyu Guan. Enhancing quality for hevc compressed videos. *IEEE Transactions on Circuits and Systems for Video Technology*, 2018.

[20] Ren Yang, Mai Xu, Zulin Wang, and Tianyi Li. Multi-frame quality enhancement for compressed video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6664–6673, 2018.

[21] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep

residual channel attention networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 286–301, 2018.

[22] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9308–9316, 2019.