Motion-aware deep video coding network

Khan, Rida, Liu, Ying

Rida Khan, Ying Liu, "Motion-aware deep video coding network," Proc. SPIE 11395, Big Data II: Learning, Analytics, and Applications, 113950B (21 April 2020); doi: 10.1117/12.2560814

Event: SPIE Defense + Commercial Sensing, 2020, Online Only, California, United States

Motion-Aware Deep Video Coding Network

Rida Khan, Ying Liu*
Department of Computer Science and Engineering, Santa Clara University
Santa Clara, CA 95053, USA

ABSTRACT

Recent advances in deep learning have achieved great success in fundamental computer vision tasks such as classification, detection and segmentation. Nevertheless, the research effort in deep learning-based video coding is still in its infancy. State-of-the-art deep video coding networks explore temporal correlations by means of frame-level motion estimation and motion compensation, which require high computational complexity due to the frame size, while existing block-level inter-frame prediction schemes utilize only the co-located blocks in preceding frames, which did not consider object motions. In this work, we propose a novel motion-aware deep video coding network, in which inter-frame correlations are effectively explored via a block-level motion compensation network. Experimental results demonstrate that the proposed inter-frame deep video coding model significantly improves the decoding quality under the same compression ratio.

Keywords: artificial intelligence, convolution neural network, computer vision, deep learning, motion prediction, structural similarity index, video coding, video compression

1. INTRODUCTION

Online videos now account for 80% of all web traffic. With this abundant increase in video data all around and the rise of the technology to stream these videos, it is becoming increasingly important to compress and process this video data efficiently. Machine learning is bringing in new solutions in almost every industry which now includes issues relating to image and video compression. Traditional video codecs have been in place for a while, however, advances in machine learning is gearing up to bring a change in that. Due to the increased amount of video content that is being created and uploaded by users, a high transmission bandwidth is desired for streaming videos. Machine learning is providing effective algorithms to reduce the bit rate while maintaining the quality of video streaming. One key issue is to train a network that predicts a frame from neighboring frames despite the object motions. In this work, we propose a deep learning-based video compression framework which uses both intra- and inter-frame correlations. First, we train an intra-frame compression network to compress all the odd frames. Second, we train a prediction network that predicts each even frame by the decoded preceding and next odd frames, implicitly exploring inter-frame motions. Finally, a residue compression network is trained to compress the residue between the target even frame and its prediction. We adopt block-level processing to reduce the computational complexity. We showcase our results in terms of the structural similarity (SSIM) index between the decoded frames and the ground truth for different compression ratios.

The rest of the paper is organized as follows: Section 2 focuses on the existing models and methods being used for video compression while Section 3 elaborates on our proposed model in detail. Section 4 showcases the experimental setup along with the results of the models tested on commonly used video sequences. Finally, Section 5 concludes our paper.

2. RELATED WORK

In conventional video coding [1], there are two approaches: intra-frame compression and inter-frame compression. While intra-frame compression only considers data redundancy within a single frame, inter-frame compression takes into consideration the correlation among successive frames, uses the previous or future reference frames to predict the current frame, and only compresses the residue between the target frame and the prediction. Since residual data contains much less energy than the original video frame, encoding it requires much less resources and hence the coding efficiency can be improved.

*Corresponding author. Email: yliu15@scu.edu.

Big Data II: Learning, Analytics, and Applications, edited by Fauzia Ahmad,
Proc. of SPIE Vol. 11395, 113950B · © 2020 SPIE · CCC
code: 0277-786X/20/$21 · doi: 10.1117/12.2560814

Recently, deep learning methods, especially the convolutional neural networks (CNN), have been introduced showcasing a new direction for image and video compression [2]. For image compression, the method in [3] adopted CNN as a pre-processing scheme to reduce the original image to a smaller-scale representation, followed by traditional image compression/decompression, then an additional CNN performs super-resolution to reconstruct the image at the original scale. For video coding, most works adopted conventional video coding concepts in their deep learning frameworks. In [4]-[6], several modules in conventional video codecs such as intra-frame prediction, residue coding, motion estimation, motion vector encoding and decoding, and motion compensation are replaced by CNN modules. The method in [7] realized scalable video coding by using iterative residue coding and generated a sequence of coarse-to-fine predictions of the target image block.

Besides, existing deep learning-based video coding schemes can be categorized into frame-based and block-based schemes. For frame-based video coding, a spatial-temporal prediction through CNN is adopted in [8] as a post-processing scheme to enhance the quality of versatile video coding (VVC) compressed videos. The method in [9] also performed post-processing to enhance the quality of intra-frame coding, using a multi-stage progressive generative adversarial network (GAN). In [6], motion prediction and compensation were performed between the current frame and the reconstructed previous frame, through CNN structures. In [10], an IBBBIBBB coding structure is adopted, in which the I-frames were encoded and decoded using a regular CNN, and the intermediate B frames were generated by hierarchical interpolation. The method in [11] is based on GAN, and proposed a latent-space linear interpolation for inter-frame prediction. The generator then produces the corresponding pixel-domain frames from the interpolated latent variables. For these aforementioned schemes, since the entire frame is predicted, the computational complexity is high. For block-based video coding, in [7], the co-located blocks from decoded previous frames and the decoded neighbor blocks in the current frame are extracted as the reference blocks and serve as the input of a prediction network, hence both inter-frame and intra-frame prediction is utilized; in [12], co-located blocks from the decoded previous four frames were extracted as the reference blocks and serve as the input of a generative network to extrapolate the target block in the current frame. In these two methods, although the complexity is lower than predicting the entire frame, only co-located blocks from the reference frames were used, which ignores the motion among successive frames.

3. PROPOSED NETWORK

The proposed framework involves three convolutional neural networks: an intra-frame compression network, an inter-frame motion-aware prediction network and a residue compression network. The overall architecture is shown in Fig. 1. We consider a video sequence as a series of frames: $T = \{F_1, F_2, \dots, F_N\}$, where N is the number of frames in that sequence. As shown in Fig. 1, if we're predicting a frame F_t for some time slot t then we first compress the previous and next frame F_{t-1} and F_{t+1} respectively for some compression ratio. Then the decoded frames $\hat{F}_{t-1}$ and $\hat{F}_{t+1}$ become the input of the prediction network. This prediction network uses the previous and next decoded frames to generate a prediction F_t^P for the current frame F_t. This prediction is subtracted from the original target frame F_t to get the residue. The residue is then compressed and reconstructed using the residue network to get $\hat{F}_t^r$. The final decoding $\hat{F}_t$ is the addition of the prediction and the decoded residue. Table 1 shows the input and output shape and configuration in each network layer for the compression net and the residue compression net. Table 2 shows the structure of the prediction net.

3.1 Compression net

The first network is a compression network which encodes and decodes each frame for some compression ratio, as shown in Fig. 2. The input to this network would be some set of frames $T = \{F_1, F_2, \dots, F_N\}$ and the output would be the set of decoded frames $\hat{F}_1$, $\hat{F}_2$, ..., $\hat{F}_N$. This network is block-based where the input and output block size are both $16 \times 16 \times 3$. The encoder input is a $16 \times 16 \times 3$ block with the red, green, and blue color channels, which is compressed to a certain final shape depending on the compression ratio used. The filters used, as can be seen in Table 1, are 5×5 filters which capture features of each frame by convolutions. The decoder input is the new shape $\frac{16}{S_1} \times \frac{16}{S_2} \times c$ where c is the number of channels in the encoder output and S_1 and S_2 are the strides, the value of which depends on the compression ratio being used. The number of channels is one of the following $\{1, 2, 3\}$ and the strides are picked from $\{2, 4, 8\}$. The output of the decoder is a reconstructed block of size $16 \times 16 \times 3$.

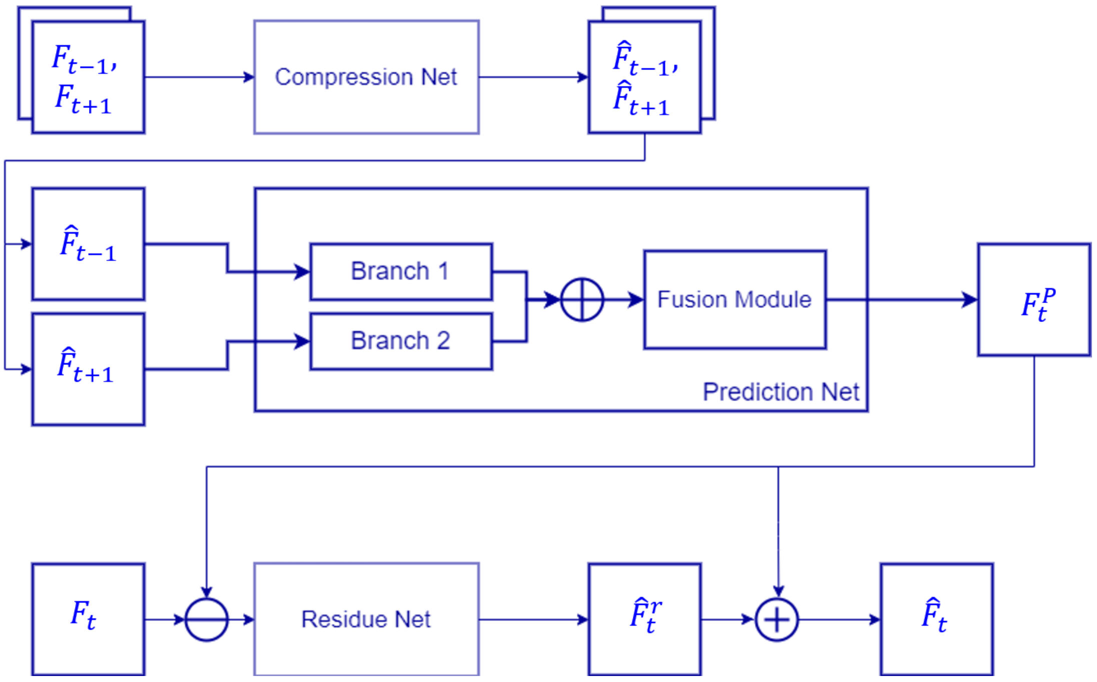


Figure 1. The architecture of the proposed motion-aware deep video coding network.

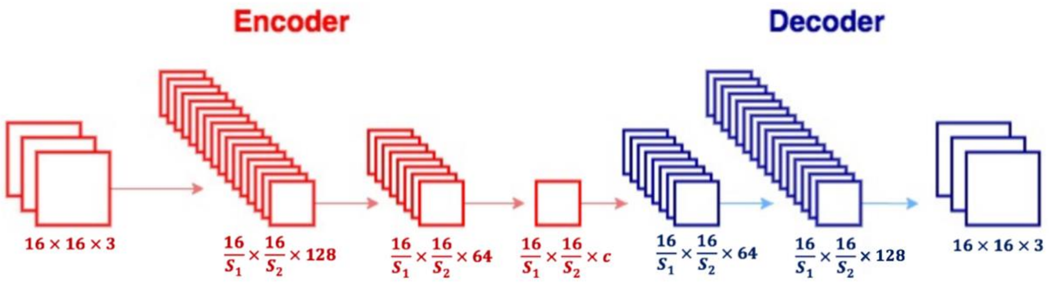

Encoder
Decoder
$16 \times 16 \times 3$
$\frac{16}{S_1} \times \frac{16}{S_2} \times 128$
$\frac{16}{S_1} \times \frac{16}{S_2} \times 64$
$\frac{16}{S_1} \times \frac{16}{S_2} \times c$
$\frac{16}{S_1} \times \frac{16}{S_2} \times 64$
$\frac{16}{S_1} \times \frac{16}{S_2} \times 128$
$16 \times 16 \times 3$

Figure 2. The architecture of the compression net and the residue net. The compressed block size is $\frac{16}{S_1} \times \frac{16}{S_2}$, where S_1 and S_2 are the strides depending on the compression ratio used. The number of channels $(c) = 1, 2$, or 3 for the encoder output. The final output of the decoder is the reconstructed block in the current frame.

Table 1. The structure of the compression net and the residue compression net for compression ratio $\frac{1}{24}$.

	Layer type/ stride	Filter shape	Output shape	Parameters #
			16x16x3 (Input Block)	
Encoder	Conv / Stride = 4	5x5	4x4x128	9,728
	Conv / Stride = 1	5x5	4x4x64	204,864
	Conv / Stride = 1	5x5	4x4x2	3,202
Decoder	TransConv / Stride = 1	5x5	4x4x64	3,264
	TransConv / Stride = 1	5x5	4x4x128	204,928
	TransConv / Stride = 4	5x5	16x16x3	9,603
	Total:			432,388

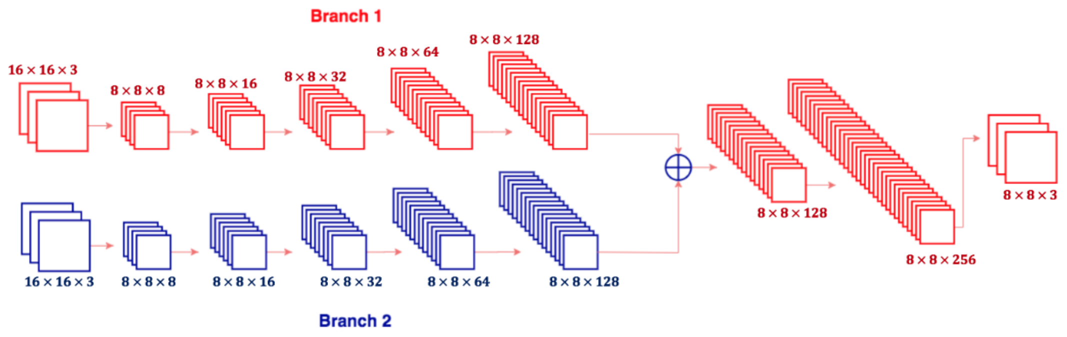


Figure 3. The architecture of the prediction net. The reference blocks of the previous and the next decoded frames predict the target block of the current frame.

Table 2. The structure of the prediction net.

	Layer Type/ Stride	Filter Shape	Output Shape	Parameters #
			16x16x3 (Input Block 1)	
			16x16x3 (Input Block 2)	
Branch 1	Conv1 / Stride = 2	5x5	8x8x8	608
	Conv2 / Stride = 1	5x5	8x8x16	3,216
	Conv3 / Stride = 1	5x5	8x8x32	12,832
	Conv4 / Stride = 1	5x5	8x8x64	51,264
	Conv5 / Stride = 1	5x5	8x8x128	204,928
Branch 2	Conv 6/ Stride = 2	5x5	8x8x8	608
	Conv7 / Stride = 1	5x5	8x8x16	3,216
	Conv8 / Stride = 1	5x5	8x8x32	12,832
	Conv9 / Stride = 1	5x5	8x8x64	51,264
	Conv10 / Stride = 1	5x5	8x8x128	204,928
	Concatenate: Conv5, Conv10		8x8x256	
Fusion	Conv / Stride = 1	5x5	8x8x128	819,328
	Conv / Stride = 1	5x5	8x8x256	819,328
	Conv / Stride = 1	5x5	8x8x3	19,203
	Total:			2,203,683

The loss function adopted to train the compression network is the mean-squared-error (MSE) as defined in equation (1) where $B_{t,m}$ is the 3D-tensor representing the mth block at time slot t which is compressed and $\hat{B}_{t,m}$ is the decoded output from the compression net. N_B is the number of pixels in the block.

$$L_{CN} = \frac{1}{3N_B}\left\|B_{t,m} - \hat{B}_{t,m}\right\|_F^2 \tag{1}$$

3.2 Prediction net

The second network is the prediction network. We aim at predicting the mth block in the tth frame $B_{t,m} \in \mathbb{R}^{8\times8\times3}$ by two larger reference blocks $\hat{B}_{t\pm1,m} \in \mathbb{R}^{16\times16\times3}$ in the decoded previous and next frame ($\hat{F}_{t-1}$ and $\hat{F}_{t+1}$), both centered at $B_{t,m}$. In this way implicit motion compensation is carried out. Fig. 3 depicts the structure of this network: it has two branches of 5 layers each. Branch 1 takes the reference block from the decoded previous frame as the input, and Branch 2 takes the reference block from the decoded next frame as the input. The strides of the first layer in Branch 1 and Branch 2 are 2.

The filters used, as can be seen in Table 2, are 5×5 filters which capture features from each reference block by convolutions. Each branch generates an intermediate output of dimension $8 \times 8 \times 128$. The two intermediate outputs are then concatenated channel-wise and processed by the final fusion module. The fusion module has 3 layers and generates the final predicted block $B_{t,m}^{P} \in \mathbb{R}^{8\times 8\times 3}$. Using such block-wise motion-aware prediction, the prediction procedure for the tth frame F_t as shown in Fig. 1 can be described as

$$F_t^P = \text{Fusion}\{\text{Branch1}(\hat{F}_{t-1}) \oplus \text{Branch2}(\hat{F}_{t+1})\}, \tag{2}$$

where $\oplus$ stands for channel-wise concatenation. The loss function adopted to train our prediction network is the mean-squared-error (MSE) as defined in equation (3) where $B_{t,m}$ is the 3D-tensor representing the original mth block in the tth frame and $B_{t,m}^P$ is the 3D-tensor representing the output of the prediction net for that block.

$$L_{PN} = \frac{1}{3N_B} \left\| B_{t,m} - B_{t,m}^P \right\|_F^2 \tag{3}$$

3.3 Residue net

The third network is the residue network which encodes and decodes the residue between the prediction of the current frame and the original current frame. It has the same encoder and decoder structure as the compression net, shown in Fig. 2. We take a set of frames $T = \{F_1, F_2, \dots, F_N\}$ and their predictions from the prediction net $P = \{F_1^P, F_2^P, \dots, F_N^P\}$ as obtained by equation (2). The input to the residue net is the residue which is calculated using equation (4) and hence, would be $R = T - P = \{F_1^r, F_2^r, \dots, F_N^r\}$. Again, this is a block-based method in which the input residue block size is $16 \times 16 \times 3$.

$$F_t^r = F_t - F_t^P \tag{4}$$

The residue goes through the residue net and is compressed to a certain shape depending on the compression ratio used. The output of the decoder is the reconstructed residue block of size $16 \times 16 \times 3$. This decoded residue is added back to the prediction as shown in equation (5) to generate the final decoding of the current frame F_t.

$$\hat{F}_t = F_t^P + \hat{F}_t^r \tag{5}$$

The loss function adopted to learn the residue network is defined in equation (6) where $B_{t,m}^r = B_{t,m} - B_{t,m}^P$ is the residue between the original mth block of the tth frame $B_{t,m}$ and its prediction $B_{t,m}^P$. This residue is compressed and $\hat{B}_{t,m}^r$ is the decoded residue block at the output of the residue net.

$$L_{RN} = \frac{1}{3N_B} \left\| B_{t,m}^r - \hat{B}_{t,m}^r \right\|_F^2 \tag{6}$$

4. EXPERIMENTAL RESULTS AND ANALYSIS

In this section, we evaluate the performance of our proposed deep video coding network on three video sequences commonly used in industrial video coding standards.

4.1 Dataset

Three video sequences: BQ Mall, Basketball Drill, and Party Scene are used in the experiment. Each frame in all three video sequences has a resolution of 480×832 pixels. For each video sequence, the first 100 frames were included in the training set. 80% of the training set was used for training and 20% for validation. The next 100 frames were used for testing.

4.2 Implementation Details

We adopted block-based processing. For the compression net and residue compression net, we divide the frames into $16 \times 16 \times 3$ blocks, and each block is individually compressed and decompressed. The compression ratios are listed in Table 3. For the prediction net, the size of the target blocks to predict is $8 \times 8 \times 3$, and the size of the reference blocks is $16 \times 16 \times 3$.

Table 3. Compression ratios used in the experiments.

Encoder Input Shape	Encoder Output Shape	Compression Ratio	
$\mathbf{16 \times 16 \times 3}$	$4 \times 4 \times 1$	$\frac{1}{48}$	0.021
$\mathbf{16 \times 16 \times 3}$	$4 \times 4 \times 2$	$\frac{1}{24}$	0.042
$\mathbf{16 \times 16 \times 3}$	$8 \times 4 \times 2$	$\frac{1}{12}$	0.083
$\mathbf{16 \times 16 \times 3}$	$8 \times 4 \times 3$	$\frac{1}{8}$	0.125
$\mathbf{16 \times 16 \times 3}$	$8 \times 16 \times 1$	$\frac{1}{6}$	0.167

4.3 Evaluation Metric

To evaluate the performance of the proposed motion-aware deep video coding network, we calculate the SSIM between the decoded frames and the ground truth frames. The SSIM is defined in equation (7) where x is the ground-truth pixel value and y is the decoded pixel value, respectively; μ_x and μ_y are the average of x and y, respectively; ${\sigma_x}^2$ and ${\sigma_y}^2$ are the variances of x and y, respectively; σ_{xy} is the covariance of x and y and finally, c_1 and c_2 are variables used to stabilize the division with a weak denominator.

$$\mathrm{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{({\mu_x}^2 + {\mu_y}^2 + c_1)({\sigma_x}^2 + {\sigma_y}^2 + c_2)} \tag{7}$$

In Figs. 4-6, we compare the perceptual quality of the decoded frames by our proposed scheme, the intra-frame compression network and the intermediate prediction net. Fig. 4 shows the results of the BQ Mall video sequence at a compression ratio of 1/12. We observe that our proposed method (Fig. 4 (d)) achieved the highest SSIM value compared to the compression net (Fig. 4 (b)) and the intermediate prediction result (Fig. 4 (c)). Similar results can be observed in Figs. 5 and 6 for the Basketball Drill and Party Scene video sequences, respectively. We also show enlarged regions of these video sequences to further emphasize the perceptual quality improvement in our results as can be seen in Figs. 7-9. With the same compression ratio, the proposed motion-aware prediction and residual coding produces (Figs. 7-9 (d)) much clearer decoded scenes with more texture details, as compared to the intra-frame compression net (Figs. 7-9 (b)) and the intermediate prediction result (Figs. 7-9 (c)).

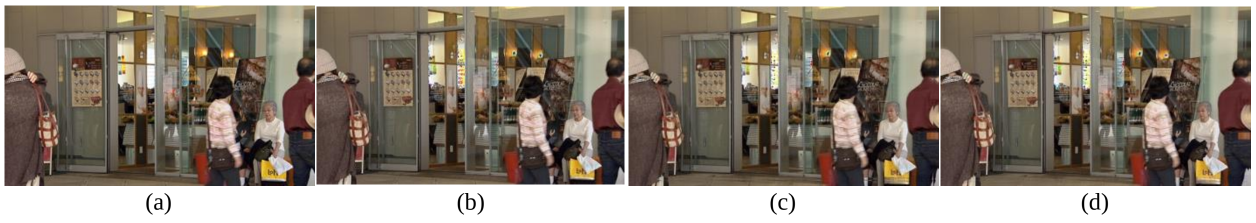


Figure 4. The perceptual quality of the BQ Mall video sequence at the compression ratio 1/12: (a) the original frame; (b) the result of the compression net with SSIM = 0.84; (c) the result of the prediction net with SSIM = 0.82; and (d) the result of the residue network which is our final decoding with SSIM = 0.89.

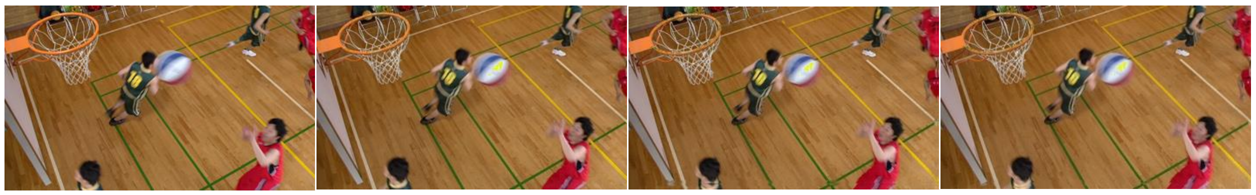

(a) (b) (c) (d)

Figure 5. The perceptual quality of the Basketball Drill video sequence at the compression ratio 1/12: (a) the original frame; (b) the result of the compression net with SSIM = 0.90; (c) the result of the prediction net with SSIM = 0.89; and (d) the result of the residue network which is our final decoding with SSIM = 0.93.

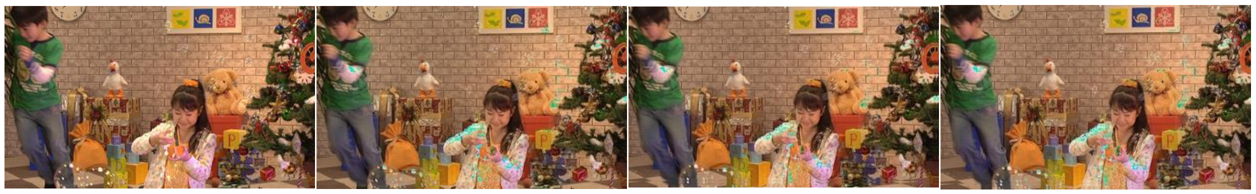

(a) (b) (c) (d)

Figure 6. The perceptual quality of the Party Scene video sequence at the compression ratio 1/12: (a) the original frame; (b) the result of the compression net with SSIM = 0.73; (c) the result of the prediction net with SSIM = 0.71; and (d) the result of the residue network which is our final decoding with SSIM = 0.84.

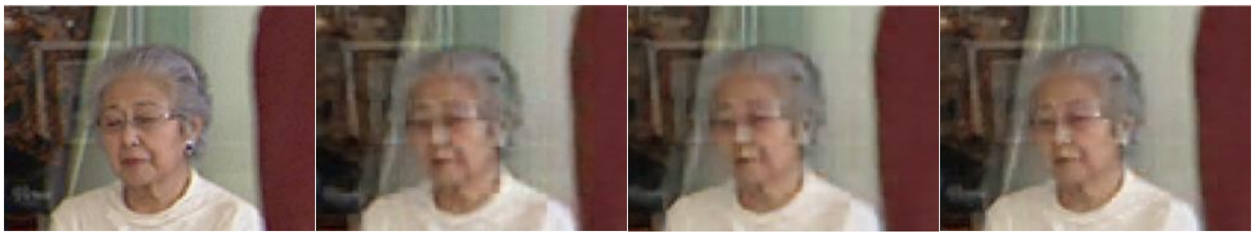

(a) (b) (c) (d)

Figure 7. The enlarged region of the BQ Mall video sequence at the compression ratio 1/12: (a) the original frame; (b) the result of the compression net with SSIM = 0.84; (c) the result of the prediction net with SSIM = 0.82; and (d) the result of the residue network which is our final decoding with SSIM = 0.89.

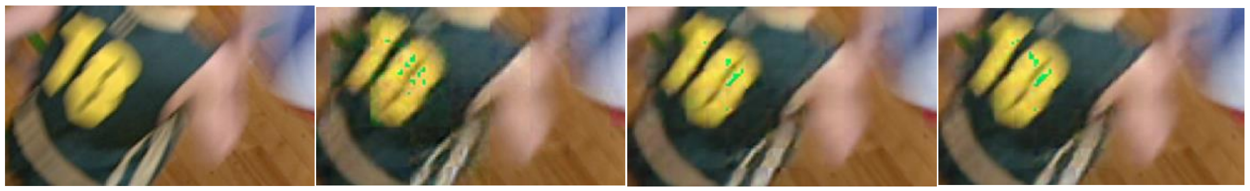

(a) (b) (c) (d)

Figure 8. The enlarged region of the Basketball Drill video sequence at the compression ratio 1/12: (a) the original frame; (b) the result of the compression net with SSIM = 0.90; (c) the result of the prediction net with SSIM = 0.89; and (d) the result of the residue network which is our final decoding with SSIM = 0.93.

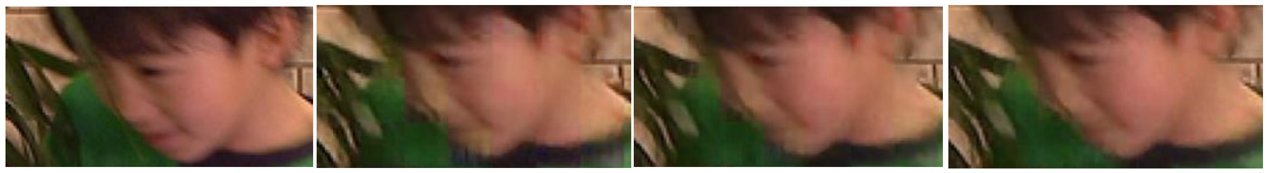
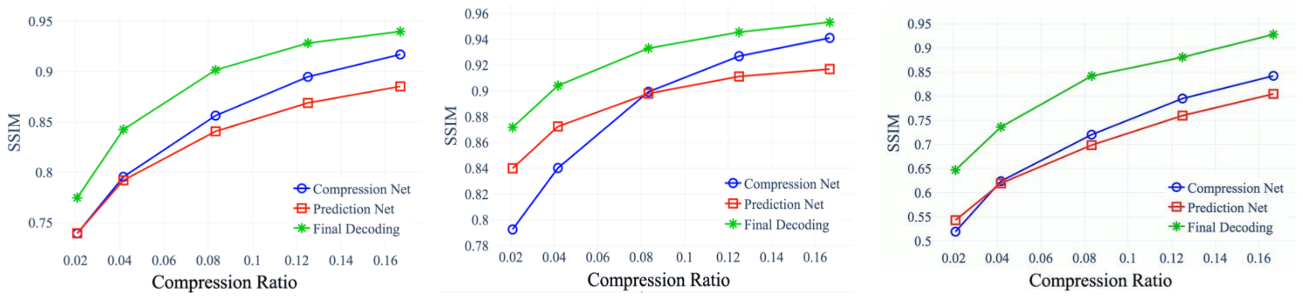

(a) (b) (c) (d)

Figure 9. The enlarged region of the Party Scene video sequence at the compression ratio 1/12: (a) the original frame; (b) the result of the compression net with SSIM = 0.73; (c) the result of the prediction net with SSIM = 0.71; and (d) the result of the residue network which is our final decoding with SSIM = 0.84.

Finally, we showcase the SSIM curves versus different compression ratios in Fig. 10 for the three video sequences. Again, it is observed that the final decoding (Final Decoding) of our proposed scheme achieves the highest SSIM values for all compression ratios.



(a) BQ Mall (b) Basketball Drill (c) Party Scene

Figure 10: The SSIM versus compression ratios for (a) the BQ Mall video sequence; (b) the Basketball Drill video sequence; and (c) the Party Scene video sequence.

5. CONCLUSION

In this paper, we propose a new deep learning framework for video frame compression and reconstruction using three networks: compression, prediction and residue network. While the odd-number frames are compressed by intra-frame coding, the proposed prediction net predicts the target blocks in even-number frames by reference blocks in the decoded previous and next odd frames. Since the reference blocks involve a larger spatial area than the target block, implicit inter-frame motion compensation is performed. Our experimental studies demonstrated the effectiveness of the proposed model, in terms of higher SSIM values and better visual quality compared to intra-frame coding and the intermediate prediction result.

REFERENCES

[1] I. E. Richardson, *The H.264 Advanced Video Compression Standard*, 2nd ed., New York: John Wiley & Sons, 2010.
[2] D. Liu, Y. Li, J. Lin, H. L, and F. Wu, “Deep learning-based video coding: a review and a case study,” *ACM Comput. Surveys*, vol. 53, no. 1, Feb. 2020.
[3] F. Jiang, W. Tao, S. Liu, J. Ren, X. Guo, and D. Zhao, “An end-to-end compression framework based on convolutional neural networks,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 10, pp. 3007-3018, Oct. 2018.
[4] T. Chen, H. Liu, Q. Shen, T. Yue, -based video compression,” in *Proc. IEEE Visual Commun. and Image Process. (VCIP)*, Saint Petersburg, FL, Dec. 2017, pp. 1-4.
[5] G. Lu, W. Ouyang, D. Xu, X. Zhang, C. Cai, and Z. Gao, “Dvc: An end-to-end deep video compression framework,” in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR)*, Long Beach, CA, June 2019, pp. 11006-11015.
[6] O. Rippel, S. Nair, C. Lew, S. Branson, A. Anderson and L. Bourdev, “Learned Video Compression,” in *Proc. IEEE Int. Conf. Comput. Vision (ICCV)*, Seoul, Korea, Oct. – Nov. 2019, pp. 3453-3462.
[7] Z. Chen, T. He, X. Jin, and F. Wu, “Learning for video compression,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 2, pp. 566-576, Feb. 2020.
[8] X. Meng, X. Deng, S. Zhu, B. Zeng. “Enhancing quality for VVC compressed videos by jointly exploiting spatial details and temporal structure,” in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Taipei, Taiwan, Sept. 2019, pp. 1193-1197.
[9] Z. Jin, P. An, C. Yang, and L. Shen, “Quality enhancement for intra frame coding via cnns: An adversarial approach,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Calgary, Alberta, Canada, Apr. 2018, pp. 1368-1372.
[10] C.-Y. Wu, N. Singhal, and P. Krahenbuhl, “Video compression through image interpolation,” in *Proc. European Conf. Comput. Vision (ECCV)*, Munich, Germany, Sept. 2018, pp. 416-431.
[11] S. Santurkar, D. Budden, and N. Shavit, “Generative compression,” in *Proc. IEEE Picture Coding Symp. (PCS)*, San Francisco, CA, USA, June 2018, pp. 258-262.
[12] J. Lin, D. Liu, H. Li, and F. Wu, “Generative adversarial network-based frame extrapolation for video coding,” in *Proc. IEEE Conf. Visual Commun. and Image Process. (VCIP)*, Taichung, Taiwan, Dec. 2018, pp. 1-4.