Enhanced video coding system with online training neural network

Paper ID 6029

Abstract

We herein propose an efficient video coding system with online training neural network to improve coding efficiency. A frame restoration convolutional neural network (FRCNN) is trained for each group of pictures of each sequence to repair the quality of each reconstructed frame. Using only the current encoding video stream as a training set, the FRCNN can restore the reconstructed frames very meticulously. Even at low bit rates, the final output of the FRCNN can improve the video quality effectively. Moreover, an efficient parameter coding scheme is applied to compress the parameters of the online training FRCNN. Subsequently, the compressed bits are transmitted to the decoder as part of the encoded bitstream. Compared with the latest High Efficiency Video Coding standard video coding, the proposed system can achieve 3.8–14.0% Bjøntegaard-Delta rate reduction, which is much higher than most of the existing neural-network-based video coding systems. The restoration network will be an additional part of the traditional standard codec without any structure change, thereby rendering it compatible with the existing coding systems.

1. Introduction

With the increasing quantity of video applications and the high demand for video quality, the development of video coding technology continues. In the past few decades, classic video codec cores including MPEG-2, H.264/AVC [1], and H.265/HEVC [2] have been developed. Currently, versatile video coding [3] is available. Newer standards are often exchanged for increased coding efficiency at the expense of increased algorithm complexity, such as more complex prediction block partitioning schemes and finer subpixel interpolation to improve prediction accuracy. However, in general, encoders are always present in the hybrid coding architecture of prediction and transformation.

Deep neural networks, especially convolutional neural networks (CNNs), have developed rapidly in computer vision and image processing, and achieved impressive results in image super-resolution, denoising, and image restoration [12]-[17]. The method to apply neural networks to codecs has aroused widespread interest. For example, image autoencoders [4] embody CNNs' ability to compress information. Because of the complex structure and maturity of the video codec technology, it is difficult to achieve gains by replacing the whole traditional standard codec directly with the network. Therefore, most of the existing solutions involve replacing individual components in the traditional encoder with neural networks.

For example, neural networks are used for coding unit partitions and intramode selection [5][6][7]. Ning Yan et al. [8] designed an interpolator based on a super-resolution CNN for fractional pixel estimation to improve the accuracy of motion estimation. WS Park et al. [9] proposed a CNN based in-loop filter called IFCNN to replace the de-blocking filter and sample adaptive offset filter in HEVC. Kang et al. [10] designed the MMS-net to achieve the same goal. In addition, Li et al. [11] developed a CNN-based block upsampling scheme for intracoding. The image blocks can be coded directly with conventional codec or downsampled first before the traditional codec and subsequently upsampled to its original resolution using the CNN or traditional methods.

To train an effective network, a large dataset is typically required. All of the existing works embedded well-trained neural networks trained with generic datasets to the video codec, which did not take advantage of containing large amount of data in video codec applications. Moreover, the current decoder cannot be utilized as is not compatible with the existing standard.

Hence, we propose to add a frame restoration convolutional neural network (FRCNN) after the traditional standard codec such that it is compatible with the existing standards. Moreover, the FRCNN is trained for each group of pictures (GOPs) in the current video. Hence, the video stream can be well fitted by the CNN, and the network can exhibit a strong repair ability. This type of operation means that we are actively performing an "overfitting" operation for the video and storing the image details lost because of the codec into the entire network. On this basis, we transmit the parameters of the restoration network as part of the code stream to the decoder. Hence, we can fully utilize the

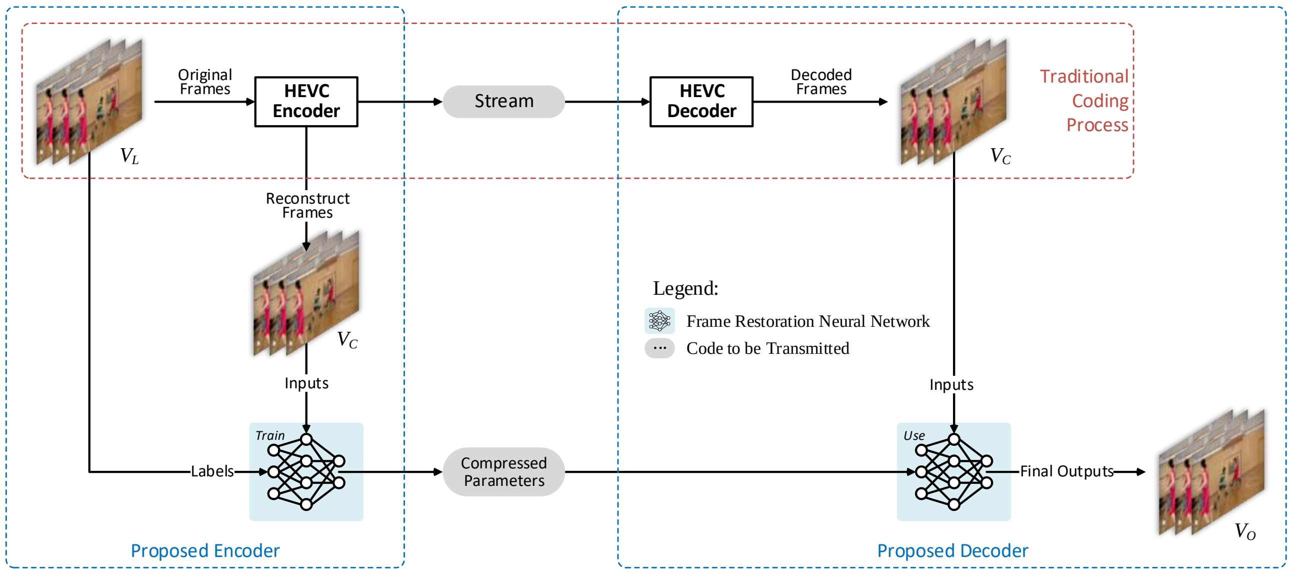


Figure 1: Processing flow of the proposed codec system. The frame restoration convolutional neural network will be an additional component of the standard encoder and will not modify the original system. The network is trained while video coding, after which the parameters of the network are transmitted as part of the code stream.

advantages in the predictive coding of traditional encoders and the detail repairing ability of the convolutional neural network.

The remainder of this paper is organized as follows. Section 2 introduces the selection and modification of the FRCNN, as well as the exploration of parameter compression methods. Section 3 presents the experimental results, followed by the conclusions in Section 4.

2. Proposed codec system

As shown in Figure 1, the FRCNN is trained while video coding is being performed, after which the parameters of the network are transmitted as part of the code stream. The task of the FRCNN is to restore the decoded frames back to the original frames by learning the missing details in the codec.

2.1. Network selection

The authors of the denoising convolutional neural network (DnCNN) [12] demonstrated the multitasking ability of the CNN; this implies that the super-resolution (SR) network, denoising network, and deblocking network are structurally consistent. Therefore, most SR networks can be used as restoration networks.

Many SR networks exhibit good results in terms of objective indicators such as peak signal-to-noise ratio (PSNR), e.g., the super-resolution convolutional neural network (SRCNN) [13] and fast super-resolution convolutional neural network (FSRCNN) [14]. Kim et al. constructed 20 layers on very deep convolutional network (VDSR) based on a visual geometry group (VGG) network with 3 × 3 small convolution kernel and used residual learning to achieve a PSNR increase of more than 1 dB [15].

Lai et al. proposed the deep Laplacian pyramid super-resolution network (LapSRN) for a fast and accurate image super-resolution [16]. They addressed only the feature maps of the corresponding size for different magnifications and extracted multiple intermediate outputs for residual learning.

Considering the requirement of transmitting the restoration network parameters as part of the code stream to the decoder, the network should contain as few parameters as possible. A deep recursive residual network (DRRN) [17] consists of several recursive blocks, in which several residual units are stacked. All the residual units in each recursive block share the same weights, thus allowing the network to maintain a smaller size while achieving more convolution layers. Hence, the DRRN requires only half the parameters of VDSR to achieve the same effect.

Therefore, we choose the DRRN as the base of the frame restoration network.

2.2. Loss function

Pixel wise mean squared error (MSE) or L2 loss is the most widely used loss function for general image restoration and is also a major performance measure (PSNR) for those problems. The original DRRN used MSE as its loss function. We record the decoded frame as V_L, and the output of the restoration network is recorded as V_O. The mean square error loss is

$$loss_{MSE} = \frac{1}{K}\sum_{i=1}^{K}\left\|V_L - V_O\right\|^2 \qquad (1)$$

where K is the sample number in a training batch. The author's research in [18] shows that the L1 loss function presents a better effect in image processing.

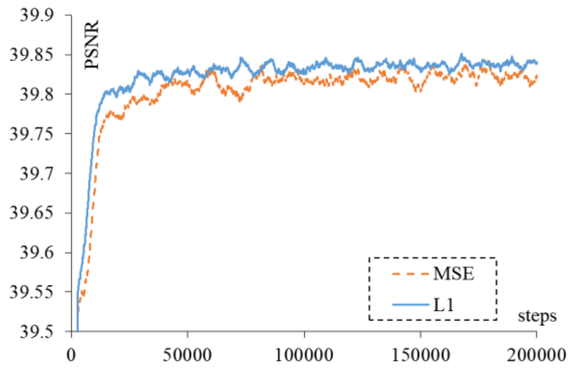


Figure 2: Different loss functions yield different repair effects. Orange indicates MSE and blue indicates L1 loss function (smoothed).

$$loss_{L1} = {}^{1}\!/_{K} \sum_{i=1}^{K} \left| V_L - V_O \right| \qquad (2)$$

As shown in Figure 2, the L1 loss function can yield some improvements and render the network easier to converge.

2.3. Remove excess BN layer

The batch normalization (BN) layer is used extensively in the original DRRN. The BN layer can normalize the intermediate output such that the subsequent layers are easier to converge. However, because the dataset is relatively small in our application, the BN layer cannot learn the mean and variance correctly, thus causing severe shocks (shown in Figure 3). Therefore, we removed most of the BN layers in the DRRN. The experimental results in Figure 4 prove that retaining the BN layer in the first convolution layer will yield the best results.

2.4. Parameter compression

To reduce the transfer file size, we propose to compress the parameters. High-precision gradient calculation is required during neural network training. To ensure a proper backpropagation, floating point numbers at least 32-bits wide are required. Generally, it is not necessary to reduce

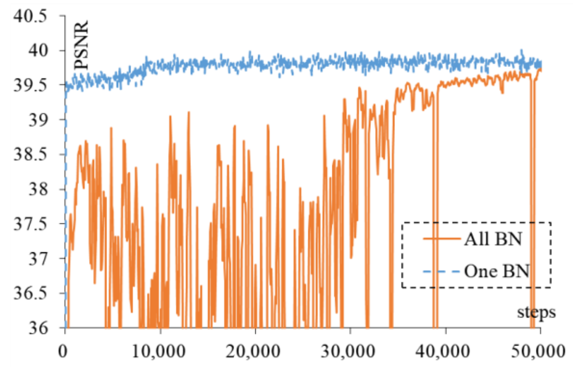


Figure 3: Excessive BN layers impede the convergence of the neural network (marked in orange, smoothed). Training is more stable after removing the additional BN layers (marked in blue).

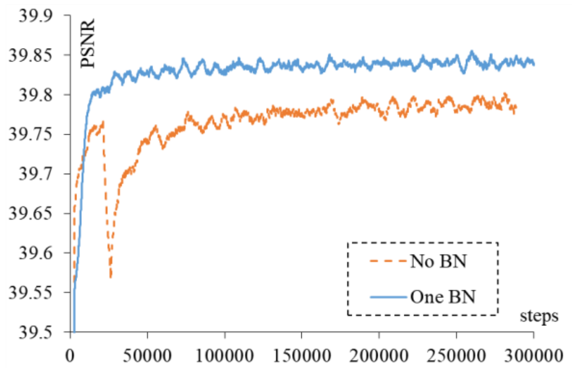


Figure 4: Retaining a BN in the first layer (marked in blue) achieves better results than removing all of them from the network (marked in orange).

high-precision parameters after the network training is completed.

The authors in [19] quantified the parameters into integers such that the parameter volume can be reduced to a quarter. However, unlike classification networks, which are insensitive to processing errors, excessive parameter accuracy loss in frame restoration networks can cause significant noise. As shown in Figure 5, the frame output by the network after parameter quantization contains visible noise.

We preserved the floating-point form of the network parameters, but reduced the bit width to 16 bits, thus reducing the size of the parameter files by half. Tests have shown that this operation does not result in deteriorated accuracy.

Simultaneously, we performed Huffman coding on the converted parameters to further reduce its volume. Consequently, the size of the parameter is reduced by ~3.5–6%.

2.5. Overall system

The proposed FRCNN reserved the structure of one recursive block and nine residual units of the DRRN (20 layers) and removed the additional BN layers.

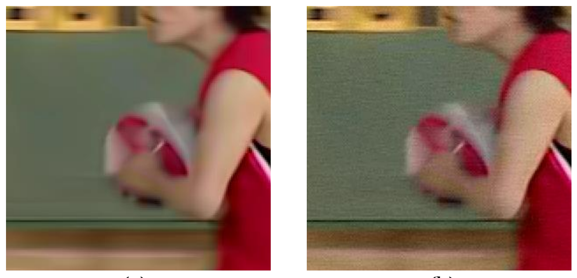


Figure 5: Figure (a) shows the normal restoration network output. Figure (b) shows the output obtained with quantified network parameters. The significant noise present in (b) indicates that parameter quantization is not suitable for compressing the FRCNN.

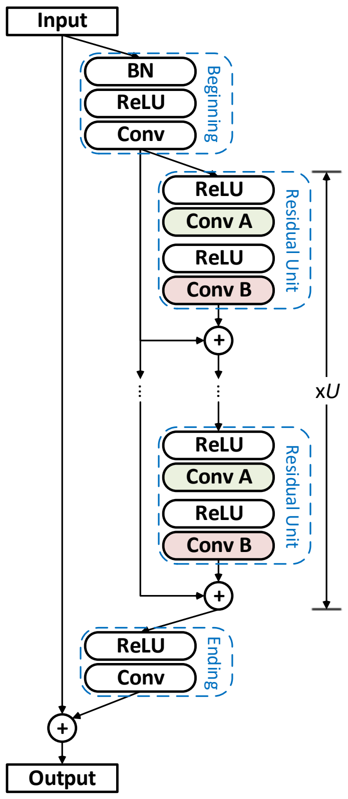


Figure 6: Structure of the frame restoration network that was eventually adopted. The number of residual units U is set to 9, similar to that in the DRRN [17]. All convolutional layers marked with the same color share the same weight. $\oplus$ is the pixel-wise addition.

The structure of the FRCNN is shown in Figure 6. Assuming that the reconstructed frame of the codec output is V_C, which will also be used as the input of FRCNN, The result of the beginning convolution block f_b can be formulated as follows:

$$f_b(V_C,\theta_b) = Conv\left(ReLU\left[BN(V_C)\right],\theta_b\right) \tag{3}$$

in which θ_b is the weight of the beginning convolution layer.

Each residual unit contains two convolutional layers, A and B. Layers A and B in all residual units share the same weights of θ_A and θ_B, respectively. Each residual unit can be expressed as

$$RU(x) = Conv\left(ReLU\left[Conv\left(ReLU[x],\theta_A\right)\right],\theta_B\right) \tag{4}$$

In the original DRRN, a recursive block contains one beginning convolutional layer and several stacked residual units whose weights are shared. Considering that the optimal structure will always contain only one recursive block, this concept will not be incorporated herein. The output of the residual units stack is expressed as

$$RUS(x,\theta_A,\theta_B) = \underbrace{x + RU\left(\cdots x + RU(x)\right)}_{U} \tag{5}$$

Sequence	Frame rate (fps)
BasketballDrive	50
BQTerrace	60
Cactus	50
Kimono1	24
ParkScene	24

Table 1. Video streams in HEVC test set of Class B.

The result of the ending convolution block f_e can be formulated as

$$f_e(x,\theta_e) = Conv\left(ReLU[x],\theta_e\right) \tag{6}$$

Further, the final output of the residual network is as follows:

$$V_O = f_e\left(RUS\left[f_b(V_C,\theta_b),\theta_A,\theta_B\right],\theta_e\right) + V_C \tag{7}$$

The number of residual units U is set to 9. Therefore, this network contains 20 convolution layers. The other hyperparameters that can be adjusted include the following:

1. **Perform a network fitting training for every *N* frames.** At every N frames, the parameters of the FRCNN will be updated and transmitted to the decoder. Generally, N is equal to the GOP size in video encoding.

2. **The number of channels *M* of the FRCNN network.** More channels represent more parameters and the stronger fitting ability of the network. However, with the increase in M, the additional parameters typically yield less gain.

3. **The quantization parameter (QP) of the encoder.** The gain of the network is different under different QPs.

3. Experiments and analysis

The experiments were based on the H.265/HEVC video encoder application library x265 [20], version 2.6. We implemented the proposed online training neural network system with TensorFlow 1.10 [21] and trained them using NVIDIA GeForce GTX 1080Ti on Ubuntu 16.04 LTS.

As shown in Table 1, five high-definition resolution (1920 × 1080) sequences in the HEVC test set of Class B are selected. The GOP size was set to 50, and all of the sequences were configured to be low_delay_P. The x265 encoder's preset was set to medium, and other options were chosen with default values. Four networks of QPs 25, 28, 30, and 35 were trained for each sequence. The trained parameters were subsequently compressed and sent to the decoder together with the encoded bitstreams.

Unlike the traditional video encoder, once the network structure is fixed, the number of parameters is constant. The larger the QP, the worse is the encoding quality, and hence a smaller encoded bitstream file. The size of the network should also be reduced to avoid an overload in parameter burden. Therefore, the number of network channels for the first three QPs is 64 while that of 35 QP is 32. The frame quality is measured using the PSNR.

Methods	PSNR(dB)		Size of the encoded bitstream (Mb)	
	value	Δ	value	Δ
QP25	39.49	-	16.86	-
QP24	39.83	↑0.34	21.63	↑28.28%
QP25+FRCNN+ 16bit+Huffman	39.84	↑0.35	17.94	↑6.39%

Table 2. Coding result of *BasketballDrive* with QP25.

3.1. Performance comparison

The proposed FRCNN improves the quality of the decoded frame; however, the parameter transmission results in additional code stream burden. This is similar to improving the picture quality by reducing the QP in traditional coding; hence, the code rate will increase accordingly.

To compare with the standard codec system, we select a benchmark QP and subsequently perform a comparison to determine the QP that requires an increased bit rate when the same level of picture quality is achieved. As shown in Table 2, using BasketballDrive as an example, QP25 is used to encode the first 50 frames as a reference. When the QP is set to 24, the average PSNR increases by 0.34 dB, while resulting in a 28.28% increase in the size of the encoded bitstream. In contrast, the proposed codec system can achieve the same video quality, but only yields 6.39% of additional bitstream size. The network presents significant performance advantages in this example.

As shown in Table 4, compared with the latest HEVC standard video coding, the proposed system can achieve 3.8–14.0% Bjøntegaard-Delta rate (BD-rate) [22] reduction, which is much higher than all of the existing neural-network-based video coding systems in Table 3. Figure 8 shows the R-D curves of five sequences. Figure 7 shows the visual comparison results. By applying the proposed system, the bit rate can be reduced significantly while maintaining the video quality.

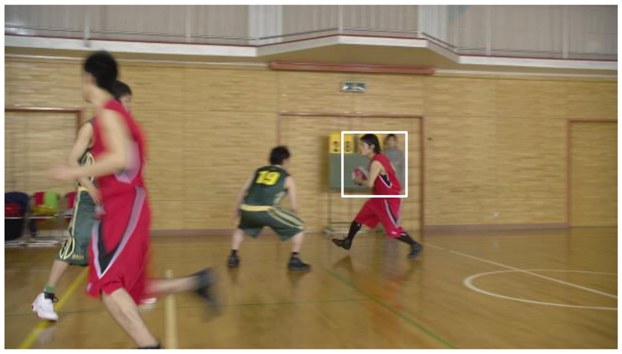
3.2. Hyperparameter setting

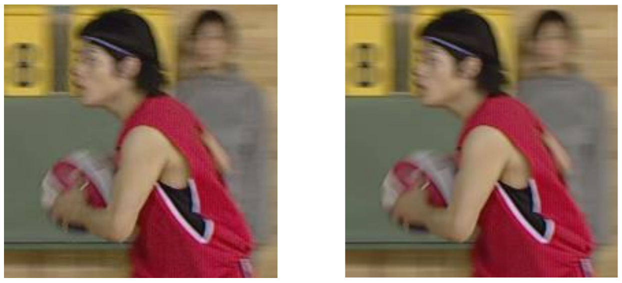
We can change the number of parameters by adjusting the number of channels in the network. Using more channels can yield a higher PSNR of the repaired frame. However, the relationship between them is nonlinear, and different videos present different situations. The specific number of channels that each video should match under different coding configurations cannot be generalized. Here we discuss the hyperparameter setting in following two aspects:

1. When the network structure is fixed, the higher the QP is set, the more obvious is the repair effect of the network. In addition, when the QP is large, the bit stream file generated by the conventional encoder is small, such that the proportion of the network parameters can be extremely large. Therefore, we can use fewer channels for large QPs.

2. As the number of channels increases, the PSNR gain due to channel increments becomes smaller. Therefore, a small number of channels will yield a better BD curve.

Consequently, less channels are beneficial for improving the BD-rate performance; however, a minute PSNR gain is meaningless. To ensure that each network can yield at least 0.3 dB of PSNR gain to the frame, we set up 64 channels for QP25, QP28, and QP30, which is a statistical balance of bit stream volume and repair effects. When the QP is 35, 64


QP 25: PSNR = 39.49dB Bits = 16.86Mb


QP 24

PSNR(Avg.) = 39.82dB
Bits = 21.63Mb(↑ 28.28%)

This work with QP25

PSNR(Avg.) = 39.84dB
Bits = 17.94Mb(↑ 6.39%)

Figure 7: Visual Comparison of the proposed video coding system and the standard HEVC/H.265 video coding.

Methods	Configuration	Compatible with the standard	BD-rate
CNNIF[8]	Low-Delay P	No	-0.9
IFCNN[9]	Low-Delay P	No	-2.8
	Random Access		-2.6
	All Intra	Yes	-4.8
MMS-net[10]	All Intra	No	-8.5
Up-sampling[11]	All Intra	No	-5.5
Our proposed	Low-Delay P	Yes	-10.76

Table 3. Comparing with related neural network based video coding algorithms

Sequence	QP	x265 [20]		This work		BD-Rate
		Bitrate (kbps)	PSNR (dB)	Bitrate (kbps)	PSNR (dB)	
BasketballDrive	25	17682.0960	39.4914	18812.2150	39.8445	-14.068997
	28	9643.5360	38.6153	10773.6550	39.0772	
	30	7122.3920	38.0642	8252.5110	38.6188	
	35*	3582.1520	36.3313	3877.2238	36.9879	
BQTerrace	25	58171.2384	37.9653	59527.3812	38.3985	-20.150092
	28	27057.7728	36.0173	28413.9156	36.6685	
	30	15916.0608	34.9889	17272.2036	35.5920	
	35*	4460.1216	32.4214	4814.2078	33.1762	
Cactus	25	26402.2720	38.1311	27532.3910	38.4919	-9.987860
	28	12097.4400	36.8853	13227.5590	37.3377	
	30	8475.0960	36.0872	9605.2150	36.6145	
	35*	3868.0640	33.7890	4163.1358	34.3097	
Kimono1	25	7216.2125	41.5925	7758.6696	42.1573	-5.766042
	28	4797.5731	40.6632	5340.0302	41.2335	
	30	3802.7981	39.9419	4345.2552	40.5392	
	35*	1982.4346	37.6755	2124.0690	38.1844	
ParkScene	25	9843.6096	39.0897	10386.0667	39.5734	-3.831738
	28	5804.8781	37.4694	6347.3352	37.9554	
	30	4221.5770	36.4305	4764.0341	36.9282	
	35*	1800.2650	33.7234	1941.8994	34.0780	

* When the QP is 35, the network contains only 32 channels (C32).

Table 4. Comparison with the HEVC standard.

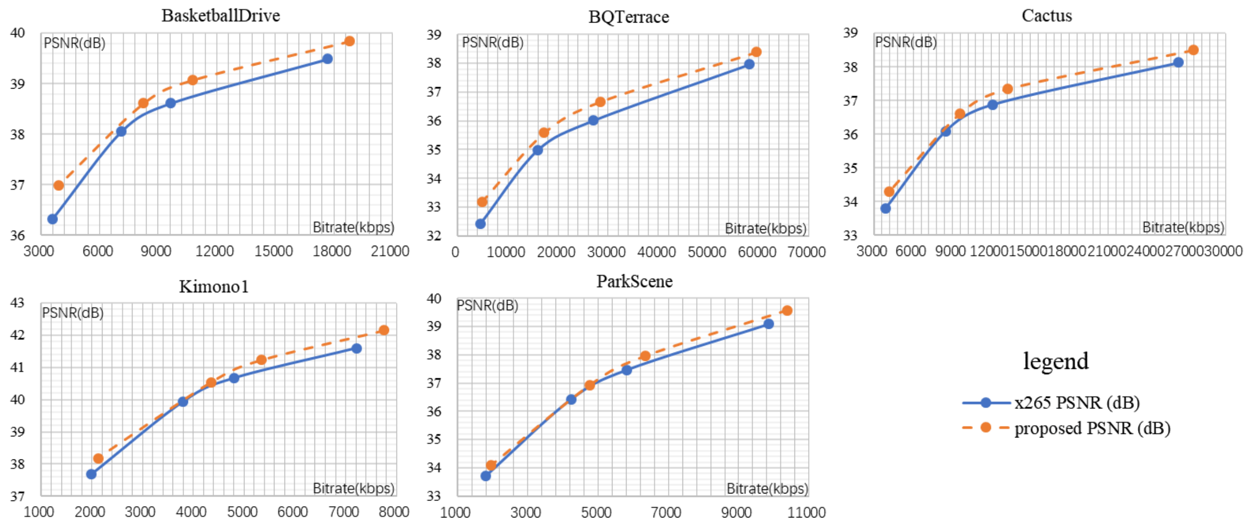


Figure 8: R-D curves to compare with HEVC standard video coding.

channels yield a large bit rate burden (especially for Kimono1 and ParkScene); therefore, we used 32 channels instead.

4. Conclusion

We herein proposed an efficient video coding system with online training neural network to improve coding efficiency. The training process of the neural network is equivalent to extracting and recording information from the training set, which is the same as data compression. We applied this concept to video coding. By attaching a frame restoration network to the traditional encoder to learn the lost details of the decoded frame during training, and storing them in the network parameters, we built an additional residual "CNN video encoder". This restoration network was trained only on the currently encoded, and its parameters were compressed and transmitted to the decoder together with the code stream generated by the traditional encoder after the training was completed. Compared with the latest HEVC standard video coding, the proposed system could achieve 3.8–14.0% BD-rate reduction, which is much higher than most of the existing neural-network-based video coding systems. The restoration network is an additional part of the traditional standard codec without any structure, thus rendering it compatible with the existing coding systems.

References

[1] Wiegand, T., Sullivan, G. J., Bjontegaard, G., & Luthra, A. (2003). Overview of the H.264/AVC video coding standard. *IEEE Transactions on circuits and systems for video technology*, *13*(7), 560-576.

[2] Sullivan, G. J., Ohm, J., Han, W. J., & Wiegand, T. (2012). Overview of the high efficiency video coding (HEVC) standard. *IEEE Transactions on circuits and systems for video technology*, *22*(12), 1649-1668.

[3] Beyond HEVC: Versatile Video Coding project starts strongly in Joint Video Experts Team. [Online]. Available: https://news.itu.int/versatile-video-coding-project-starts-strongly, accessed on: Aug. 4, 2018

[4] Theis, L., Shi, W., Cunningham, A., & Huszár, F. (2017). Lossy image compression with compressive autoencoders. *arXiv preprint arXiv:1703.00395*.

[5] Laude, T., & Ostermann, J. (2016, December). Deep learning-based intra prediction mode decision for HEVC. In *2016 Picture Coding Symposium (PCS)* (pp. 1-5). IEEE.

[6] Yu, X., Liu, Z., Liu, J., Gao, Y., & Wang, D. (2015, September). VLSI friendly fast CU/PU mode decision for HEVC intra encoding: Leveraging convolution neural network. In *2015 IEEE International Conference on Image Processing (ICIP)*(pp. 1285-1289). IEEE.

[7] Liu, Z., Yu, X., Chen, S., & Wang, D. (2016, May). CNN oriented fast HEVC intra CU mode decision. In *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*(pp. 2270-2273). IEEE.

[8] Yan, N., Liu, D., Li, H., & Wu, F. (2017, May). A convolutional neural network approach for half-pel interpolation in video coding. In *2017 IEEE international symposium on circuits and systems (ISCAS)* (pp. 1-4). IEEE.

[9] Park, W. S., & Kim, M. (2016, July). CNN-based in-loop filtering for coding efficiency improvement. In *2016 IEEE 12th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP)* (pp. 1-5). IEEE.

[10] Kang, J., Kim, S., & Lee, K. M. (2017, September). Multi-modal/multi-scale convolutional neural network based in-loop filter design for next generation video codec. In *2017 IEEE International Conference on Image Processing (ICIP)* (pp. 26-30). IEEE.

[11] Li, Y., Liu, D., Li, H., Li, L., Wu, F., Zhang, H., & Yang, H. (2018). Convolutional neural network-based block up-sampling for intra frame coding. *IEEE Transactions on Circuits and Systems for Video Technology*, *28*(9), 2316-2330.

[12] Zhang, K., Zuo, W., Chen, Y., Meng, D., & Zhang, L. (2017). Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, *26*(7), 3142-3155.

[13] Dong, C., Loy, C. C., He, K., & Tang, X. (2016). Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, *38*(2), 295-307.

[14] Dong, C., Loy, C. C., & Tang, X. (2016, October). Accelerating the super-resolution convolutional neural network. In *European conference on computer vision* (pp. 391-407). Springer, Cham.

[15] Kim, J., Kwon Lee, J., & Mu Lee, K. (2016). Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1646-1654).

[16] Lai, W. S., Huang, J. B., Ahuja, N., & Yang, M. H. (2018). Fast and accurate image super-resolution with deep laplacian pyramid networks. *IEEE transactions on pattern analysis and machine intelligence*.

[17] Tai, Y., Yang, J., & Liu, X. (2017). Image super-resolution via deep recursive residual network. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition* (pp. 3147-3155).

[18] Zhao, H., Gallo, O., Frosio, I., & Kautz, J. (2017). Loss functions for image restoration with neural networks. *IEEE Transactions on Computational Imaging*, *3*(1), 47-57.

[19] Han, S., Mao, H., & Dally, W. J. (2015). Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*.

[20] x265 HEVC Encoder / H.265 Video Codec project [Online]. Available: http://www.x265.org/, accessed on: Nov. 12, 2018

[21] Tensorflow Branch r1.10 [Online]. Available: https://www.tensorflow.org/versions/r1.10/, accessed on: Nov. 12, 2018

[22] Bjontegaard, G. (2001). Calculation of average PSNR differences between RD-curves. *VCEG-M33*.