RESIDUAL GUIDED DEBLOCKING WITH DEEP LEARNING

Wei Jia, Li Li, Zhu Li

University of Missouri Kansas City

Xiang Zhang, Shan Liu

Tencent America

ABSTRACT

The block-based coding structure in hybrid video coding framework inevitably introduces compression artifacts such as blocking, ringing etc. Recently, neural network based loop filters are proposed to enhance the reconstructed frame. But the coding information has not been full utilized in the design of neural networks. Therefore, in this paper, we propose a Residual-Reconstruction-based Convolutional Neural Network (RRCNN) to improve the coding efficiency to its full extent, where the residual frame is fed into the network as a supplementary input to the reconstructed frame. In essence, the residual signal can provide effective information about block partitions and can help in recognizing smooth, edge and texture regions in a picture, such that more adaptive parameters can be trained to handle different texture characteristics. In addition, the network structure has been carefully designed in the proposed dual-input network to learn useful context information from two signals with their distinct features. To the best of our knowledge, this is the first work that employs the residual signal in the CNN-based in-loop filter for video coding. The experimental results show that our proposed RRCNN approach leads to significant BD-rate savings compared to HEVC and the state-of-the-art CNN-based schemes, indicating residual signal plays an important role in the enhancement of reconstructed video frames.

Index Terms— Convolutional Neural Network, High Efficiency Video Coding, In-loop Filter, Reconstruction, Residual.

1. INTRODUCTION

Advanced Video Coding (H.264/AVC) [1], High-Efficiency Video Coding (H.265/HEVC) [2] are existing popular video coding standards and Versatile Video Coding (VVC) [3] is the emerging next-generation standard under the development of Moving Pictures Expert Group (MPEG) and Video Coding Experts Group (VCEG). All of these video coding standards adopt the hybrid coding frameworks, where the major procedures include prediction, transform, quantization, and entropy coding. In the hybrid coding framework, a video frame is partitioned into non-overlapping coding blocks or units, forming the concepts of coding units (CU), prediction unit (PU), transform unit (TU) etc. Block-based coding scheme is hardware-friendly and easy to implement and it enables useful coding functionalities such as parallelization.

However, block-wise operation inevitably introduces discontinuity at the block boundaries, such that blocking artifacts are one of the major distortions [4] in video coding. Beyond that, a coarse quantization is another major factor in causing video quality degradation especially at the regions with sharp edges, which is known as the ringing artifacts. This type of ripple phenomena induces low visual quality and bad user experience [5]. In view of these, extensive Convolutional Neural Network (CNN) based in-loop filters [6–8] have been proposed to compensate artifacts and distortions in video coding. However,only the reconstruction is used as the input in most current CNN based methods. The other information in the bit stream has not been utilized to improve the performance of the in-loop filter.

To improve the performance of the CNN-based loop filter to its full extent, we propose a Residual-Reconstruction-based Convolutional Neural Network (RRCNN) utilizing both the reconstruction and residual as input. The major contributions of this work are two-folds:

- First, we employ the residual signal as the supplementary information and feed it into the neural nets in pair with the reconstructed frame. The residual is utilized as an indicator to direct the CNN learning how to augment signal missed by block-wise video compression schemes. To the best of our knowledge, this is the first work to utilize the residual signal in the purpose of in-loop filter for video coding.
- Second, the network structure is carefully designed for the dual-input CNN to fully utilize the underlying features in different input channels, where residual blocks are used for Residual-Network and a hierarchical autoencoder network is used for Reconstruction-Network.

We organize the remainder of this paper as follows. Section 2 introduces the proposed RRCNN approach. In section 3, we report and analyze the experimental results. Finally, Section 4 summarizes this paper.

2. PROPOSED ALGORITHM

In this section, we will introduce the proposed RRCNN method in detail including the architecture of the RRCNN

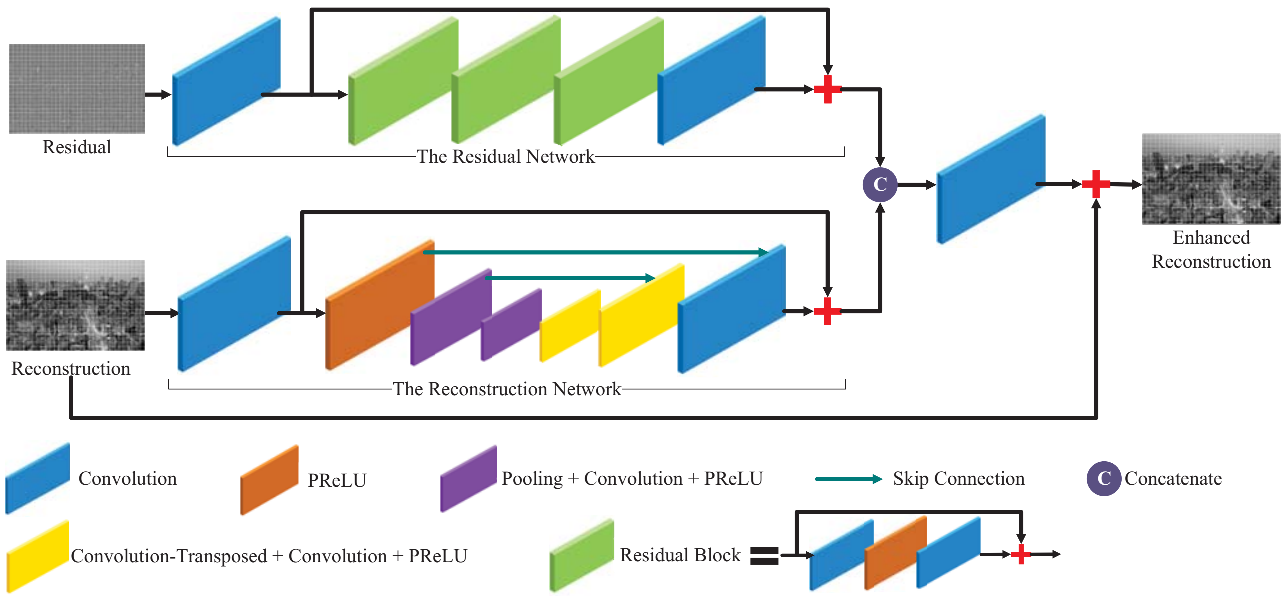

Fig. 1. RRCNN with sub-networks of both the Residual Network and the Reconstruction Network. Residuals into the Residual Network to provide the TU partition information and the detailed textures information. The Residual Network relies on residual blocks to learn features effectively with difference learning. We feed the reconstruction into the Reconstruction Network. The Reconstruction Network executes the downsampling and upsampling strategy to patch up the local and global information. This enhances reconstruction quality and aids with the difference learning approach.

Table 1. Reconstruction Network Parameters of Conv And Transposed Conv Layers

Type of Layer	Conv1	Conv2	Conv3	Transposed Conv1	Conv4	Transposed Conv2	Conv5	Conv6
Kernel Size	3×3	3×3	3×3	2×2	3×3	2×2	3×3	3×3
Feature Map Number	32	64	128	64	64	32	32	32
Stride	1	1	1	2	1	2	1	1
Padding	1	1	1	0	1	0	1	1

and the design of the Residual and Reconstruction networks.

2.1. Architecture of the proposed RRCNN framework

Fig. 1 shows the overall architecture of the proposed RRCNN framework. The proposed RRCNN framework including two sub-networks: the reconstruction network and the residual network. The reconstruction network uses the reconstruction as input and derives reconstruction feature map from the input. The residual network uses the residual as input and derives residual feature maps from the input. Note that we use different sub-networks for the reconstruction and residual as they have completely different characteristics. The residual is more sensitive while the reconstruction consisting of residual and prediction contains more global information. The feature maps derived from the two sub-networks are then concatenated together and used as the input of the last convolutional layer. In addition, we learn the difference between the input and the label to accelerate the training process.

2.2. Design of the Residual Network

We develop a Residual Network consisting of several residual blocks [9] to adapt to the features of the residual. The residual block could effectively keep the residual features and the gradient information on the shallow layers. Therefore, the proposed Residual Network can derive the distinct features from the residual frame. Considering the complexity, we use only 8 convolutional layers to derive the residual features.

In Fig. 1, the upper rectangle with dotted line shows the detailed architecture of our proposed Residual Network. The Residual Network includes three residual blocks consisting 6 convolutional layers, and two convolutional layers at the beginning and end. For each convolutional layer, we set the Kernel Size as 3×3, the Feature Map Number as 32, Stride as 1, and Padding as 1. As the Parametric Rectified Linear

Unit (PReLU) [10] has been demonstrated to be more effective than the ReLU, we employ it as the activation function in the Residual Network.

2.3. Design of the Reconstruction Network

We develop a Reconstruction Network containing several downsampling and upsampling pairs to learn the reconstruction features. The Reconstruction Network adopts the classic autoencoder architecture [11] [12] and the skip connection concatenating the encoder and decoder parts [13]. In this way, the reconstruction network is able to recover the global information and details as much as possible.

In Fig 1, the lower rectangle with dotted line shows the detailed structure of our proposed Reconstruction Network. We adopt the pooling and transposed convolutional layer to perform downsampling and upsampling, respectively. Table 1 shows the detailed configurations. For the convolutional layers, we set the Kernel Size as 3×3, Stride as 1, Padding as 1, Feature Map Number as 32, 64 or 128. For the transposed convolutional layers [14], we set the Kernel Size as 2×2, Stride as 2, Padding as 1, Feature Map Number as 64 or 32.

2.4. Loss function, dataset and training

Loss function. We employ Mean Squared Error (MSE) [15] as the loss function for our proposed RRCNN as follows,

$$L(\Theta) = \frac{1}{N}\sum_{i=1}^{N} ||\Upsilon(Y_i|\Theta) - X_i||_2^2 \qquad (1)$$

where Θ encapsulates the whole parameter set of the network containing weights and bias and $\Upsilon(Y_i|\Theta)$ denotes the network module. X_i is an original frame, where i indexes each frame. Y_i is the corresponding reconstruction, that is compressed by HEVC when we turn off its deblocking and SAO. N is the number of frames.

Dataset. We employ the DIV2K [16] [17] comprising 800 training images and 100 validating images of $2k$ resolution as the original frames. Then a modified HEVC reference software is used to encode original frames to generate the reconstruction and residual with $QP22$, $QP27$, $QP32$, and $QP37$, respectively. We finally extract 64×64 blocks from the Luma component of the reconstructed, residual, and original frames and use them as the inputs and labels for training our proposed RRCNN. In total, there are $522,939$ groups of inputs and labels for training and $66,650$ groups for validation.

Training. We train the $QP37$ model first, and then fine tune the $QP37$ model to get all the other models. We set the batch-size as 16 and the base learning rate as $1e^{-4}$. We adopt the Adaptive Moment Estimation (Adam) [18] algorithm with the momentum of 0.9 and the weight decay of $1e^{-4}$. We train the $QP37$ model with 120 epochs. After 100 epochs, we decrease the learning rate by 10 times. After the $QP37$ model is derived, We fine tune it with 20 epochs to obtain the other models. Finally, we obtain the models for all the QPs for testing.

Table 2. Performance comparison of VRCNN and RRCNN against HEVC under All Intra configuration

Class	Sequence	VRCNN vs. HEVC	RRCNN vs. HEVC
Class A	Traffic	-8.1%	**-10.2**%
	PeopleOnStreet	-7.7%	**-9.4**%
Class B	Kimono	-5.9%	**-8.6**%
	ParkScene	-6.2%	**-8.1**%
	Cactus	-2.7%	**-5.8**%
	BasketballDrive	-5.2%	**-7.7**%
	BQTerrace	-2.9%	**-4.2**%
Class C	BasketballDrill	-10.6%	**-13.8**%
	BQMall	-7.3%	**-9.3**%
	PartyScene	-4.6%	**-5.6**%
	RaceHorses	-5.8%	**-7.1**%
Class D	BasketballPass	-7.6%	**-9.5**%
	BQSquare	-5.3%	**-6.3**%
	BlowingBubbles	-5.5%	**-6.7**%
	RaceHorses	-8.9%	**-10.2**%
Class E	FourPeople	-10.0%	**-12.8**%
	Johnny	-9.1%	**-12.5**%
	KristenAndSara	-9.4%	**-11.8**%
Summary	Class A	-7.9%	**-9.8**%
	Class B	-4.6%	**-6.9**%
	Class C	-7.1%	**-8.9**%
	Class D	-6.8%	**-8.2**%
	Class E	-9.5%	**-12.4**%
Avg.	All	-6.8%	**-8.9**%

3. EXPERIMENTAL RESULTS

The proposed algorithm is implemented in HM-16.19 to compare with HEVC, VRCNN [6], and the state-of-the-art CNN-based loop filter with partition and reconstruction as input [7]. We test all the sequences defined in the HM-16.19 common test condition (CTC) [19] except for the two 10bit sequences. The all intra configuration is used to demonstrate the effectiveness of the proposed algorithm as the proposed algorithm is designed for the intra frame. We test QPs from 22 to 37 defined in the CTU to verify the performance of the proposed algorithm from low bitrate to high bitrate cases. The BD-rate is employed for a fair comparison among the performance of various methods.

3.1. Performance comparison with HEVC and VRCNN

Table 2 shows the comparison of the proposed RRCNN with VRCNN, as well as HEVC with Deblocking Filter (DF) [20]

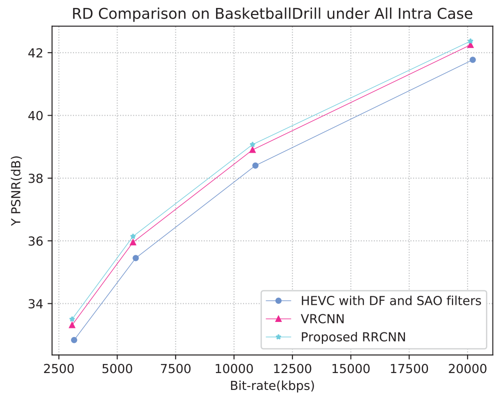

Fig. 2. Comparison of RD curves between HEVC with DF and SAO, VRCNN and proposed RRCNN on BasketballDrill.

Table 3. Computational complexity of VRCNN and RRCNN against HEVC under All Intra Case

Approches	Encoding Time(CPU)	Encoding Time(GPU)
VRCNN	172.57%	102.84%
RRCNN	243.43%	109.49%

and Sample Adaptive Offset (SAO) [21] filters on CTC test sequences under all intra configuration. We can see that RRCNN achieves a better performance with an average of 8.9% on BD-rate compared with HEVC with DF and SAO filters. For the Class E sequences, the performance improvement can be as high as 12.4%. In addition, compared with the VRCNN, the proposed RRCNN can lead to an average of 2.1% performance improvement. The RRCNN method shows consistent gains compared with VRCNN for all the tested sequences. The experimental results demonstrate that introducing the residual for augmenting the reconstruction and designing the customized network depending on input characteristics could achieve bitrate savings.

Fig. 2 shows one typical example of the Rate-Distortion (RD) curves comparison of proposed RRCNN approach, VRCNN, and HEVC with DF and SAO filters on Luma component. We can see that the PSNR of the proposed RRCNN method is higher than both VRCNN and HEVC with in-loop filters under each QP. The RD curve also demonstrates the effectiveness of the proposed algorithm.

The encoding complexities of VRCNN and the proposed RRCNN are shown in Table 3. The VRCNN complexity overhead is 172.57% while the RRCNN complexity overhead is 243.43% compared to HEVC. But they all can be improved by GPU acceleration to a similar level as HEVC.

Table 4. Performance comparison between the proposed RRCNN, the state-of-the-art method with the reconstruction and partition as input, and the method with reconstruction as the only input

Class	Partition Reconstruction vs. Reconstruction	Residual Reconstruction vs. Reconstruction
A	−0.4%	**-1.0%**
B	−0.2%	**-0.9%**
C	−0.4%	**-1.1%**
D	−0.4%	**-0.8%**
E	−0.6%	**-1.6%**
Avg.	−0.4%	**-1.0%**

3.2. Performance comparison with state-of-the-art method

Table 4 shows the performance comparison between the proposed method and the method with only construction as input, and the state-of-the-art method with partition and reconstruction as input. We can see that the proposed method saves an average of 1.0% bitrate compared with the method with only Reconstruction as the input. The experimental results demonstrate that the residual is a good supplement to the reconstruction and can bring additional benefit to improve the performance of the CNN-based loop filter.

In addition, compared to the dual-input method of the Partition+Reconstruction, the proposed dual-input approach of the Residual+Reconstruction saves an average of 0.6% BD-rate. The experimental results demonstrate that the residual is a more effective factor compared with the partition when used as the other inputs of the network. The residual can not only provide the partition information that is the same as the partition but also introduce more abundant texture information that is beneficial for the overall coding performance.

4. CONCLUSION

In this paper, we propose the RRCNN algorithm utilizing both residual and reconstruction as input to improve the performance of the in-loop filter. To the best of our knowledge, this is the first work that uses the residual to enhance the reconstruction for improving the quality of reconstructed video. In addition, we design specified networks for the residual and reconstruction utilizing their specific characteristics to further improve the performance. The proposed algorithm is implemented in the High Efficiency Video Coding (HEVC) reference software. The experimental results show the proposed algorithm can bring 8.9% performance improvement compared with HEVC, indicating the effectiveness of the proposed algorithm.

5. REFERENCES

[1] Thomas Wiegand, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra, "Overview of the h. 264/avc video coding standard," *IEEE Transactions on circuits and systems for video technology*, vol. 13, no. 7, pp. 560–576, 2003.

[2] 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*, vol. 22, no. 12, pp. 1649–1668, 2012.

[3] Benjamin Bross, Jianle Chen, and Shan Liu, "Versatile Video Coding (Draft 4)," Document ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 JVET-M1001-v6, Marrakech, MA, Jan. 2019.

[4] Shuiming Ye, Qibin Sun, and Ee-Chien Chang, "Detecting digital image forgeries by measuring inconsistencies of blocking artifact," in *2007 IEEE International Conference on Multimedia and Expo*. IEEE, 2007, pp. 12–15.

[5] Hyungjun Lim and HyunWook Park, "A ringing-artifact reduction method for block-dct-based image resizing," *IEEE transactions on circuits and systems for video technology*, vol. 21, no. 7, pp. 879–889, 2011.

[6] Yuanying Dai, Dong Liu, and Feng Wu, "A convolutional neural network approach for post-processing in hevc intra coding," in *International Conference on Multimedia Modeling*. Springer, 2017, pp. 28–39.

[7] Xiaoyi He, Qiang Hu, Xiaoyun Zhang, Chongyang Zhang, Weiyao Lin, and Xintong Han, "Enhancing hevc compressed videos with a partition-masked convolutional neural network," in *2018 25th IEEE International Conference on Image Processing (ICIP)*. IEEE, 2018, pp. 216–220.

[8] 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.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[11] Xi Chen, Diederik P Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel, "Variational lossy autoencoder," *arXiv preprint arXiv:1611.02731*, 2016.

[12] Geoffrey E Hinton and Ruslan R Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[13] 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*. Springer, 2015, pp. 234–241.

[14] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Robert Fergus, "Deconvolutional networks.," in *Cvpr*, 2010, vol. 10, p. 7.

[15] Dennis Wackerly, William Mendenhall, and Richard L Scheaffer, *Mathematical statistics with applications*, Cengage Learning, 2014.

[16] Andrey Ignatov, Radu Timofte, et al., "Pirm challenge on perceptual image enhancement on smartphones: report," in *European Conference on Computer Vision (ECCV) Workshops*, January 2019.

[17] Eirikur Agustsson and Radu Timofte, "Ntire 2017 challenge on single image super-resolution: Dataset and study," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 126–135.

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

[19] Karl Sharman and Karsten Suehring, "Common test conditions," Document ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 JCTVC-AE1100, San Diego, US, Apr. 2018.

[20] Andrey Norkin, Gisle Bjontegaard, Arild Fuldseth, Matthias Narroschke, Masaru Ikeda, Kenneth Andersson, Minhua Zhou, and Geert Van der Auwera, "Hevc deblocking filter," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1746–1754, 2012.

[21] C.-M. Fu, C.-Y. Chen, and C.-Y. Tsai, "CE13: Sample Adaptive Offset with LCU-Independent Decoding," ITU-T/ISO/IEC JCT-VC Document JCTVC-E049, Mar. 2011.