IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 17, 2020, accepted November 23, 2020, date of publication November 26, 2020,
date of current version December 9, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3040751

Offset-Based In-Loop Filtering With a Deep Network in HEVC

SO YOON LEE[1], YOONMO YANG[1], (Graduate Student Member, IEEE), DONGSIN KIM[1], SEUNGHYUN CHO[2], AND BYUNG TAE OH[1], (Member, IEEE)

[1]School of Electronics and Information Engineering, Korea Aerospace University, Goyang 10540, South Korea
[2]Department of Information and Communication Engineering, Kyungnam University, Changwon 13557, South Korea

Corresponding author: Byung Tae Oh (byungoh@kau.ac.kr)

This work was supported in part by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of ICT under Grant NRF-2019R1F1A1063229, and in part by the GRRC Program of Gyeonggi Province (Study on 3D point cloud processing and application technology) under Grant 2020-B02.

ABSTRACT With the great flexibility and performance of deep learning technology, there have been many attempts to replace existing functions inside video codecs such as High-Efficiency Video Coding (HEVC) with deep-learning-based solutions. One of the most researched approaches is adopting a deep network as an image restoration filter to recover distorted compressed frames. In this paper, instead, we introduce a novel idea for using a deep network, in which it chooses and transmits the side information according to the type of errors and contents, inspired by the sample adaptive offset filter in HEVC. A part of the network computes the optimal offset values while another part estimates the type of error and contents simultaneously. The combination of two subnetworks can address the estimation of highly nonlinear and complicated errors compared to conventional deep- learning-based schemes. Experimental results show that the proposed system yields an average bit-rate saving of 4.2% and 2.8% for the low-delay P and random access modes, respectively, compared to the conventional HEVC. Moreover, the performance improvement is up to 6.3% and 3.9% for higher-resolution sequences.

INDEX TERMS Convolutional neural network (CNN), deep learning, in-loop filter, sample adaptive offset (SAO), HEVC, video compression.

I. INTRODUCTION

With the continuous development of electronic devices such as smartphones and digital TVs, video plays increasingly important roles in our lives. At the same time, because of the rapid development of network technology, the amount of video traffic in the network is rapidly increasing. Besides an increase in the number of videos, the video resolution is becoming larger because of the increased size of the display devices. Therefore, there is heightened demand for video compression technology that maintains high quality with fewer bits.

At this writing, the most recent standard video codec is High Efficiency Video Coding (HEVC) [1]. Although its encoding and decoding processes are similar to those of previous standard codecs, such as H.264/AVC [2], it is much more efficient and faster because of its advanced algorithm and parallel processing. In detail, with the same video resolution, HEVC requires 40–50% fewer bits than H.264/AVC to yield similar-quality video. It is composed of dozens of modules, such as motion compensation, discrete cosine transform (DCT), quantization, in-loop filtering, and context-adaptive binary arithmetic coding (CABAC) [3]. Among them, we focused on in-loop filtering in this study. The in-loop filter is designed to reduce compression artifacts caused by block-wise quantization and at the same time achieve bit-rate savings by restoring a damaged frame as close as possible to the original. It consists of a deblocking filter (DF) [4] and a sample adaptive offset (SAO) filter [5]. DF reduces blocking artifacts caused by block-wise processing for the coding unit (CU). On the other hand, SAO reduces ringing artifacts that occur by reducing high-frequency components during quantization.

Recently, deep-learning approaches have attracted much research attention [6]. They have exhibited superior performance in many areas, including conventional computer vision tasks such as classification [7], and conventional image processing tasks such as image restoration and super resolution [8], [9]. As an early work in image restoration with deep learning, the super-resolution convolutional neural network

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Wei.

(SRCNN) [8] used a simple structure, but its result was impressive. The very deep super resolution convolutional network (VDSR) highly improved the performance by deep residual network architecture [9]. Inspired by the success of SRCNN and VDSR, many researchers have attempted to apply deep-learning techniques to reduce compression artifacts. As a simple modification of SRCNN, Yu *et al.* first proposed a network for removal of still image compression artifacts, called the artifacts reduction convolutional neural network (ARCNN) [10]. ARCNN directly targets JPEG [11] artifact removal, but its structure is similar to that of SRCNN. To further improve its performance, the deep dual-domain convolutional neural network (DDCN) [12] was proposed to handle the characteristics of DCT for JPEG using two separate networks for the pixel and DCT domains.

Likewise, it is believed that deep-learning approaches can be a good solution to reduce video compression artifacts, and researchers have attempted to improve video compression performance by applying deep learning in various ways. For example, a deep-learning-based network can replace conventional modules or functions, such as intra prediction, motion estimation, and in-loop filtering.

As mentioned previously, we mainly focused on in-loop filtering in this study. We propose a new network that replaces the in-loop filter, in particular the SAO of HEVC, both in the encoder and the decoder. Unlike previous deep-learning-based in-loop filtering methods that recover the distorted frames only, the proposed scheme computes and transmits offsets as side information that significantly improves the prediction accuracy of filtering with the slight sacrifice of additional bits.

The rest of this paper is organized as follows. We review related works in Section II. Section III describes the motivation and the details of the proposed method. Section IV discusses how we trained the network and the data we used for training. The results of the trained network and the analysis are discussed in Section V. Finally, we conclude this paper in Section VI.

II. RELATED WORKS

There are two ways to apply the filter-based methods in HEVC to improve coding efficiency. One is applying the filter to decompressed frames to recover the image as close as possible to the original. The coding artifact removal method belongs to this category. Another is applying the filter to intermediate reconstructed frames. Both methods attempt to recover the reconstructed image, but there is major difference between them. The latter uses the filtered image as a reference for the prediction of other frames, whereas the former does not. Therefore, the former is called post-processing, and the latter is called in-loop filtering. For example, both DF and SAO in HEVC are in-loop filtering methods.

A. POST-PROCESSING

Many researchers have been studying the use of deep-learning technology as post-processing first, because this approach does not require modification of the encoder or decoder. In addition, post-processing for the coded image is a kind of image restoration problem. It enables researchers to easily apply a well-known deep-learning network to the codec. Inspired by SRCNN, CNN-based filtering [13] was first proposed for video compression by simply using three convolution layers to improve coding efficiency. It exhibits considerable improvement in All Intra (AI) mode, but the experiments were conducted only with low-resolution video. Moreover, their results for Low Delay P (LDP) and Random Access (RA) modes were unsatisfactory. Variable-filter-size residual-learning CNN (VRCNN) [14] suggested a more complicated network, which uses various filter sizes to match various CU sizes in HEVC. However, it is still effective on AI mode only.

To apply the deep-network approach to inter frames, such as P- and B-frames, the decoder-side scalable convolutional neural network (DSCNN) [15] was proposed; it uses different architectures for intra- and inter-coded frames. The network for intra frames, called DSCNN-I, consists of five simple convolution layers, and the network for inter frames, called DSCNN-B, also consists of five convolution layers. However, DSCNN-B uses additional inputs at each layer, and those are from DSCNN-I. This means that the output of the previous layer and the output from the layer of DSCNN-I are concatenated to form a new input format for the present layer. It mimics the conventional video codec, where an inter frame uses nearby intra frames as its reference.

Unlike those networks with complicated architecture, two networks with simpler architecture using VDSR were proposed. Li *et al.* [16] only used 20 convolutional layers, which leads on average 1.6% BD-rate reduction compared with HEVC baseline in 2017 ICIP Grand Challenge. The deep convolutional neural network-based auto decoder (DCAD) [17] was proposed to use only ten convolution layers, but the results become much better than others because of its use of a residual block structure [6]. It also shows good results not only for AI mode, but also for LDP and RA mode.

B. IN-LOOP FILTERING

In-loop filtering itself is not different from post-processing, but its effect and analysis are more complicated because it affects not only the current frame, but also the other frames that refer to the current frame. The spatial-temporal residue network (STResNet) [18] was proposed as an in-loop filter, located after SAO. It uses both spatial and temporal information during network training to make it work for intra- and inter-compressed frames, but the results showed only a small improvement. Jia *et al.* [19] proposed a content-aware CNN with multiple networks. They classified training samples with the clustering method, and trained networks for each set of clustered training samples. When incorporating the network into the HEVC, it must send an additional symbol to indicate which network is used. This process does nothing but train each network according to the characteristics of each sequence. The multi-layer deep convolutional neural network (MDCNN) [20] uses convolution layers and a symmetric

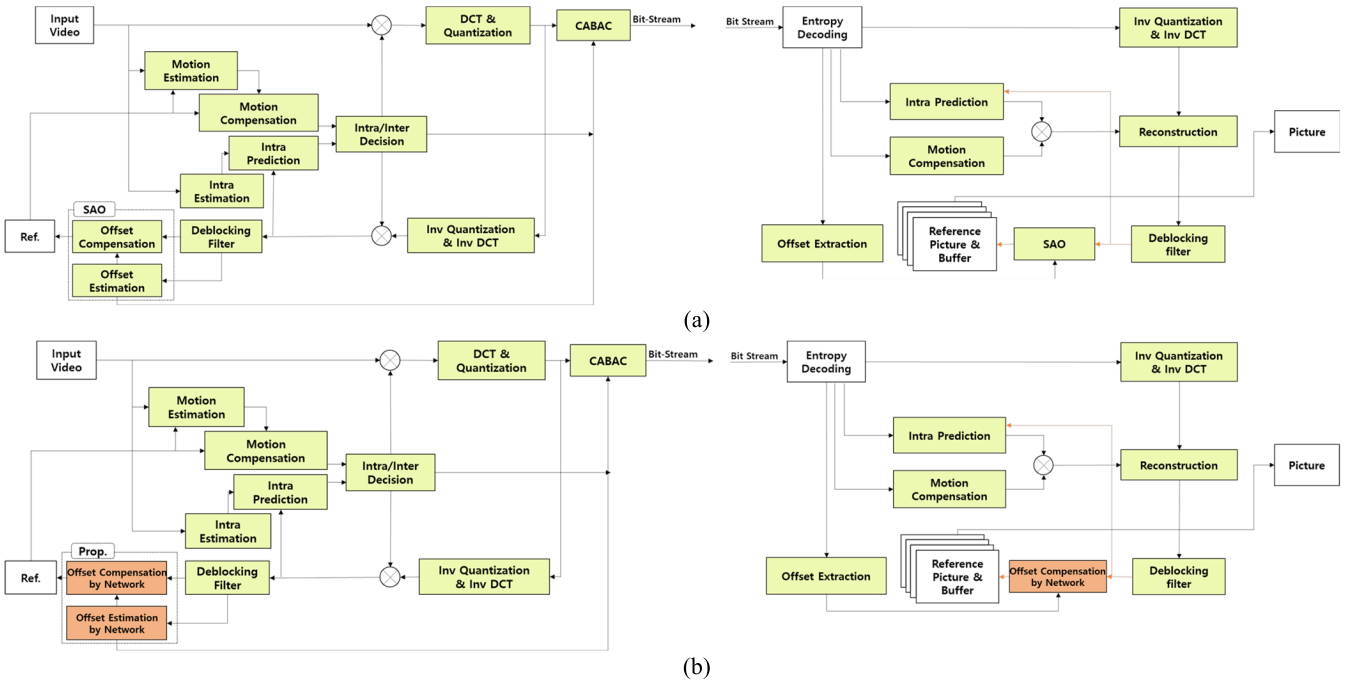

FIGURE 1. Original and modified HEVC block diagrams: (a) Original HEVC block diagram for the encoder (left) and the decoder (right); (b) modified HEVC block diagram for the encoder (left) and the decoder (right).

deconvolution layer by replacing SAO. However, it works only for AI mode, and it uses the same offsets made by SAO.

The residual highway convolutional neural network (RHNet) [21] was designed as in-loop filter, following DF and SAO. The network consists of highway units, and each of them contains three layers and identity skip connection as shortcut. The dense residual convolutional neural network (DRN) [22] was also proposed as in-loop filter. It consists of dense residual unit (DRU), where both ResNet [6] and DenseNet [23] features are used. It is embedded between DF and SAO in HEVC. The progressive rethinking network (PRN) [24] is composed of the progressive rethinking block (PRB) and the side information feature extractor (SIFE). PRB aims to keep high dimensional representative features, and SIFE aims to extract multi-scale mean value of CU as side information to improve network.

However, the in-loop filtering methods mentioned so far exhibit inferior performance improvement compared to post-processing methods, and their usage is often limited. One reason is that the state-of-the-art image restoration network design is well fitted to post-processing, whereas their simple adoption to in-loop filtering is inefficient. Instead, it is required to fully utilize a different design for in-loop filtering, and it is the starting point of our proposed scheme.

III. PROPOSED METHOD

A. RESEARCH MOTIVATION

As stated previously, HEVC has two in-loop filters: DF and SAO as in Fig. 1(a). Both filters are adopted to remove compression noise, but each targets different types of noise. An additional difference between them is the filtering scheme for noise removal. DF is turned on when the pre-defined block boundary conditions are satisfied, and the filter shape is determined according to the type of boundary conditions. On the other hand, SAO uses the explicit information of four offsets. They are computed at the encoder, and then they are sent to the decoder with CABAC [3]. Likewise, the approach of SAO uses and sends some side information to improve the estimation accuracy, and it is turned on selectively when the increase of estimation accuracy is greater than the use of additional bits. That is calculated according to the well-designed rate-distortion optimizer in HEVC.

Inspired by the overall concept of SAO, we attempted to design the network to generate and use the side information as in Fig. 1(b). The major contribution of this study is the preparation and use of offsets as side information using two deep networks. To the best of our knowledge, it is the first approach to generate and use the side information with an end-to-end deep network. Details of the network design and its application to HEVC are described in the following subsections.

B. NETWORK ARCHITECTURE

Because the proposed deep network is inspired by the SAO, it is necessary to analyze SAO in detail first. SAO is broadly composed of two parts: 1) the type of error classification for each pixel grid, and 2) the computation of the optimal offsets according to the type of error. The error classification and offset computation is conducted for every coding tree unit (CTU) independently. In the first part, the type of error can be estimated in various ways. For example, it investigates neighboring signal values, and analyzes the edge shape; this is called the SAO-edge offset (EO). Alternatively, the type of error is classified according to the pixel intensities; this

is called the SAO-band offset (BO). Once the classification is done, then the second part computes the optimal offsets for each class. Because it is complicated to determine the optimal values in terms of rate-distortion manner, it simply considers the distortion, i.e., assigning the average error value as its offset for each class. Here, it is notable that SAO acts in different ways in the encoder and the decoder. The encoder contains both the classification and the offset computation parts for SAO, whereas the decoder includes only the classification module because the optimal offsets are transmitted from the encoder.

Likewise, we decompose the SAO into two parts, and propose two separate deep networks. Each network is designed to fully mimic the concept of SAO. The first deep network is for the type of error estimation. Because SAO-EO checks the shape of the image signal along the line for the classification, the proposed error classification network (ECN) classifies the type of error signal according to neighboring image intensity values. Because ECN is also embedded at the decoder side, only the available data during decoding are used for the classification, as in SAO. To be more specific, it uses the reconstructed image after DF as input, and then it passes eight residual blocks ($R_1, R_2, \ldots, R_8$) as shown in the upper part of Fig. 2(a). Likewise, the proposed ECN can be assumed as a generalized version of SAO-EO, because the simple estimation by line-based shape for the specific direction is extended to cover any complicated relationships between the current and neighboring pixels. Moreover, it can even be assumed it also includes the concept of SAO-BO, because the pixel intensity is also considered in ECN.

The second deep network, called the offset estimation network (OEN), is designed to calculate the offsets using many pooling blocks. One proposed approach uses networks with offsets [19]. However, it uses the original offsets given by the original SAO, which strongly restricts the performance improvement with the combination of the type of error prediction network, such as ECN. Instead, we propose that the network calculates the offsets from the reconstructed image after DF and the original image. Then, OEN estimates the optimal offset values through pooling blocks to yield four offsets, where the number of offsets is adopted by SAO-EO. In addition, the offsets are set to always have positive values in SAO to save the bit rate of four offsets. Instead, the signs of the offsets are also estimated by ECN. It is notable that the original image can be used in OEN, because this network is embedded only at the encoder, whereas the four offsets are coded by CABAC, and transmitted to the decoder.

The two proposed networks, ECN and OEN, behave like those in SAO. However, there is one major difference: the order of the two modules. In SAO, the offset computation is applied when the type of error prediction is done, because the latter module assumes the suggestions of the first modules are correct. However, it is not a trivial problem for the learning-based approach, because the optimal (or correct) suggestion is not available in the practical situation. When propagating errors, for example, it is not certain whether the first and second module is responsible for the error. In the proposed architecture, instead, we propose to connect those two networks in parallel as in Fig. 2(a) and assign a penalty for both networks during error back-propagation. Then, we need not consider the intermediate ground-truth information.

Finally, the outputs of the two networks are combined at the last stage of the given system. The ECN output w has a tensor with $H \times W \times 4$, where H and W are the height and width of the given input image, respectively. The OEN output O is a 4×1 column vector including four offsets. Then, the estimated error e at (i, j) is obtained by

$$e(i,j) = \sum\nolimits_{k=1,..,4} w(i,j,k)\, O_k \tag{1}$$

If the output values of ECN are only one of ± 1 or zero, then it will select one of four offsets with sign, exactly as in SAO. The proposed method even improves this part by softening the decision for ECN to yield a value ranging from -1 to $+1$. This means that it will selectively use each of the four positive offsets according to its probability.

To train these networks, the conventional L2 loss function between reconstructed CTU and its ground-truth is used. Moreover, we consider the use of offset as an additional information by counting the number of transmitted bits. Therefore, the final loss function will be

$$L = D\left(CTU_{rec}, CTU_{gt}\right) + \lambda \cdot R\left(O_{k=1,\ldots,4}\right) \tag{2}$$

where D indicates the distortion cost by L2 function, and λ is the Lagrange multiplier used in HEVC. R measures the rate cost by counting the number bits used for each offset value as

$$R(r) = |r| + 1 \tag{3}$$

where it assumes the unary code for offset coding.

C. SYNTAX

It is necessary to adjust the syntax design for SAO, because the proposed network is slightly different from the conventional SAO. The syntax structure of the original SAO is shown in Fig. 3(a). In detail, the first flag indicates whether the SAO is turned off (SAO-Off), turned on and transmitting new offsets (SAO-New), or turned on and reusing offsets (SAO-Merge) from upper or left CTU, which is also indicated by another one-bit flag (Up/Left). Once the SAO-New mode is selected, the next flag is for the differentiation between SAO-EO and SAO-BO. After that, in the case of SAO-EO, it sends a symbol for the direction of the edge from {0°, 45°, 90°, 135°}, and four positive offsets follow. Similarly, SAO-BO sends a symbol for band position, and four offset values with signs follow.

As described previously, the proposed networks mimic the SAO, especially SAO-EO, but they have a slightly different syntax structure. Above all, they do not need a symbol for direction, because it classifies the type of error simultaneously considering all directions, or in even more complicated ways. Moreover, ECN even includes the concept of SAO-BO as well, which means the one-bit indicator for SAO-EO/BO

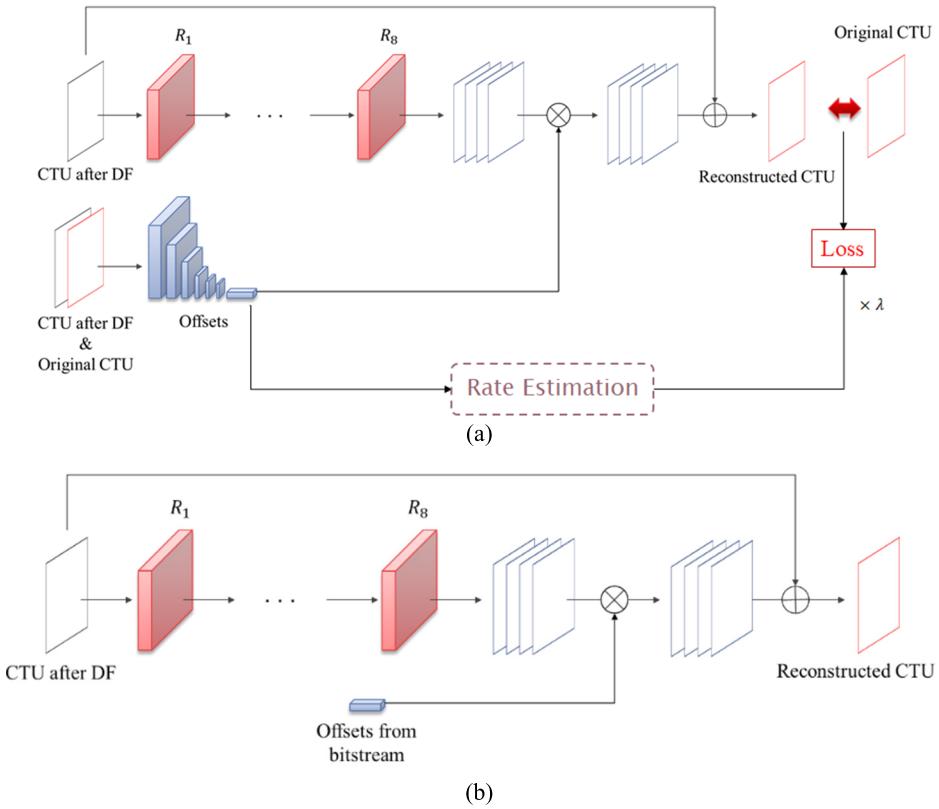

FIGURE 2. Proposed networks: (a) ECN (top) and OEN (bottom) structure at the encoder, and (b) ECN structure at the decoder.

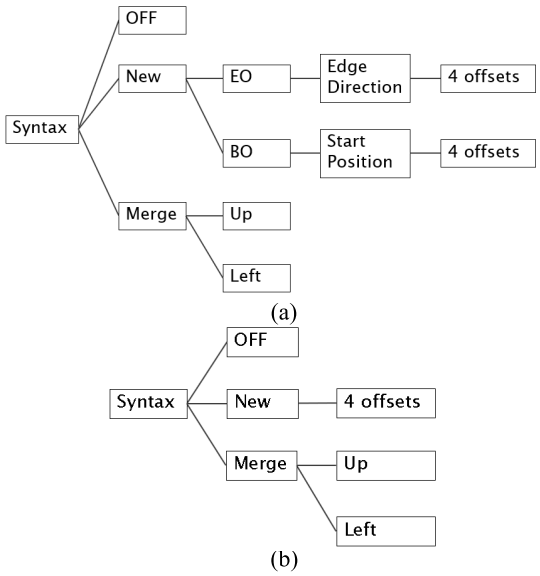

FIGURE 3. Syntax trees: (a) Syntax tree of SAO, (b) Syntax tree of the proposed method.

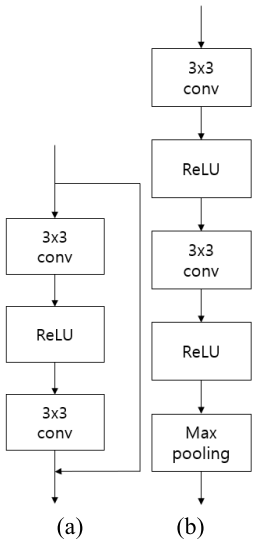

FIGURE 4. Block architecture: (a) Residual block, (b) Pooling block.

is unnecessary. These two points will save the bits for the syntax information. The summarized syntax structure of the proposed scheme is shown in Fig. 3(b).

D. BLOCKS

The proposed network uses two kinds of block structure. The first block is the residual block. The residual block consists of two convolution layers and one ReLU as seen in Fig. 4(a). In addition, it uses the residual learning method to facilitate learning. In the proposed scheme, the kernel size for whole convolution layers is set to 3×3.

The second block is the pooling block. The pooling block consists of two convolution layers, ReLU, and a max pooling layer. With the pooling layer, we can compress input data efficiently. In this part, the kernel size of the convolution layer is also set to 3×3. Likewise, the two proposed networks have relatively simple structures, but they work quite well compared to other networks with complicated

TABLE 1. Experimental results on LDP and RA modes.

Mode		Low Delay P					Random Access				
Methods		VRCNN	DCAD	DRN	RHNet	Prop.	VRCNN	DCAD	DRN	RHNet	Prop.
UHD (3840 × 2160)	*CampfireParty*	3.38	-0.21	0.36	2.03	-1.85	1.78	0.44	1.17	2.51	-1.74
	CatRobot	1.89	-1.70	-1.96	-3.77	-7.43	-4.68	-2.12	-2.82	-1.89	-6.00
	RollerCoaster	5.28	0.28	0.95	-2.07	-9.24	-0.23	1.24	1.71	4.19	-3.64
	Tango	5.12	2.01	1.51	-1.01	-8.97	-1.31	2.21	1.92	3.32	-4.10
	ToddlerFountain	-1.67	-2.78	-2.80	-3.03	-3.27	-1.56	-2.83	-3.02	-2.44	-2.13
	Average	**2.80**	**-0.48**	**-0.39**	**-1.57**	**-6.15**	**-1.20**	**-0.21**	**-0.21**	**1.14**	**-3.52**
Class A (2560 × 1600)	*NebutaFestival*	1.16	-7.74	-8.41	-7.38	-10.62	-0.62	-3.98	-3.99	-3.73	-2.16
	PeopleOnStreet	-2.02	-7.45	-8.51	-7.71	-8.83	-3.23	-7.10	-7.25	-6.09	-4.67
	SteamLocomotiveTrain	-1.18	-8.63	-9.28	-9.15	-8.14	-3.17	-3.52	-4.01	-3.30	-3.87
	Traffic	-4.26	-5.08	-5.50	-6.04	-5.40	-3.98	-3.81	-4.24	-3.96	-4.66
	Average	**-1.58**	**-7.22**	**-7.93**	**-7.57**	**-8.25**	**-2.75**	**-4.60**	**-4.87**	**-4.27**	**-3.84**
Class B (1920 × 1080)	*BasketballDrive*	2.91	-0.80	-2.52	-3.12	-4.97	0.60	0.50	-0.05	0.89	-2.96
	BQTerrace	5.02	-1.79	-2.30	-3.63	-3.08	-0.62	-2.22	-2.06	-2.85	-3.11
	Cactus	0.81	-4.39	-5.04	-5.33	-4.72	-2.91	-4.82	-5.26	-5.10	-4.38
	Kimono	5.87	-2.53	-2.99	-3.35	-6.52	0.03	-1.06	-1.28	-0.55	-2.29
	ParkScene	0.89	-1.68	-1.89	-2.00	-2.17	-1.60	-1.18	-1.72	-1.23	-2.41
	Average	**3.10**	**-2.24**	**-2.95**	**-3.49**	**-4.29**	**-0.90**	**-1.76**	**-2.07**	**-1.77**	**-3.04**
Class E (1280 × 720)	*FourPeople*	-5.43	-5.90	-6.98	-7.71	-7.09	-4.51	-6.44	-7.02	-6.98	-6.15
	Johnny	0.24	1.06	2.58	-2.71	-6.59	8.48	3.41	2.45	2.76	-4.38
	KristenAndSara	-2.14	-2.09	-1.50	-4.50	-5.33	1.47	-0.92	-0.87	-1.36	-4.64
	Average	**-2.44**	**-2.31**	**-1.97**	**-4.98**	**-6.34**	**1.81**	**-1.32**	**-1.82**	**-1.86**	**-5.05**
Average for high-resolution seqs.		**0.47**	**-3.06**	**-3.31**	**-4.40**	**-6.26**	**-0.76**	**-1.97**	**-2.24**	**-1.69**	**-3.86**
Class C (832 × 480)	*BasketballDrill*	6.83	2.23	-0.06	-0.52	1.61	2.61	-0.09	-0.41	-1.01	0.40
	BQMall	2.31	0.08	-1.29	-2.29	-0.69	0.70	-0.57	-1.15	-1.90	-0.87
	PartyScene	5.19	1.83	2.02	1.65	0.56	3.59	2.19	2.62	1.36	-0.25
	RaceHorses	1.07	-2.87	-3.08	-3.04	-3.31	-1.10	-2.71	-2.63	-2.59	-2.45
	Average	**3.85**	**0.32**	**-0.60**	**-1.05**	**-0.46**	**1.45**	**-0.29**	**-0.40**	**-1.04**	**-0.79**
Class D (416 × 240)	*BasketballPass*	3.67	2.69	1.07	0.56	0.42	2.65	1.00	0.50	-0.10	0.13
	BlowingBubbles	2.43	1.30	-0.27	-0.51	1.04	2.86	0.96	0.52	0.00	0.01
	BQSquare	10.93	9.40	5.67	4.52	1.22	9.47	4.28	4.86	3.58	-0.20
	RaceHorses	-2.29	-2.85	-3.76	-3.55	-0.22	-2.25	-3.29	-3.43	-3.29	-1.50
	Average	**3.69**	**2.63**	**0.68**	**0.25**	**0.61**	**3.18**	**0.74**	**0.61**	**0.05**	**-0.39**
Average for low-resolution seqs.		**3.77**	**1.48**	**0.04**	**-0.40**	**0.08**	**2.32**	**0.22**	**0.11**	**-0.49**	**-0.59**
Total Average		**1.57**	**-1.55**	**-2.19**	**-3.07**	**-4.15**	**0.27**	**-1.24**	**-1.46**	**-1.29**	**-2.77**

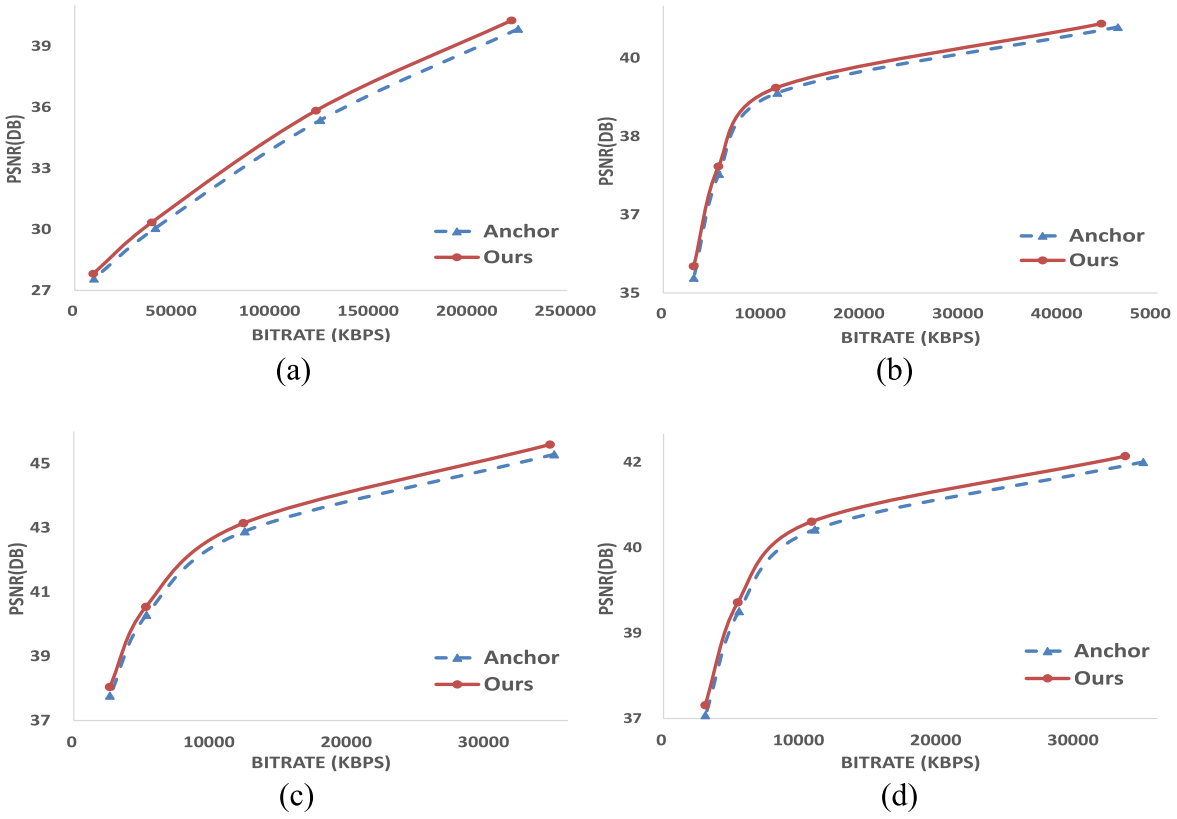

FIGURE 5. Performance comparisons for different sequences in LDP mode: (a) NebutaFestival, Class A; (b) PeopleOnStreet, Class A; (c) RollerCoaster, UHD; (d) Tango, UHD.

structures [13], [14], [18], [19]. For more details, please refer to the source code: https://github.com/yym064/DeepSAO.

IV. EXPERIMENTAL RESULTS

A. EVALUATION MEASURES

For the evaluation of the proposed method, we chose the commonly used MPEG sequences with class A, B, C, D, E, and UHD resolution, where the first 50 frames of each sequence are tested [25]. We tested on LDP and RA modes, and used four quantization parameter (QP) values (22, 27, 32, and 37), which is a common selection. To see and compare the objective results, the Bjøntegaard delta bit-rate (BDBR) [26] was calculated, which shows the average bit-rate saving with the same quality.

For a fair comparison, we chose VRCNN [14], DCAD [17], DRN [22], and RHNet [21] as competitors. VRCNN and DCAD are originally proposed as post-processing method, and RHNet and DRN are proposed as in-loop filter. All methods were applied to slightly different positions, and therefore we followed the training and testing as the original works suggested. In details, VRCNN replaces both DF and SAO, and training samples were prepared with DF and SAO turned off. On the other hand, RHNet, DCAD, and DRN are applied with both DF and SAO turned on. Both RHNet and DCAD are applied after SAO, while DRN are positioned between DF and SAO. The proposed method replaces only SAO, and therefore the training samples were obtained with only SAO turned off. Originally, RHNet and DRN were tested as in-loop filter, but they could not yield any positive gain in the common HEVC configurations. We believe the experiments in their paper were outdated. Therefore, we use them as post processing filter for comparison.

B. NETWORK TRAINING

For a fair comparison, all comparison methods are trained on the same training set with the same HEVC configuration and on the same platform. We used 37 video sequences [27]–[29] with 1920 × 1080 HD resolution to train the network. These 37 sequences were selected outside the MPEG test set; the MPEG sequences were used for testing only. We used from the second to the eleventh frames for training because the first frames of LDP or RA compressed videos are intra-compressed frames. We trained each network for LDP and RA modes, each with I-, P-, and B-frames.

The network was implemented with Pytorch [30], and HM-16.9 software [31] was used for the HEVC reference software in our experiments. All training sequences were separately compressed for each QP. The loss was minimized using the Adam optimizer [32] during back propagation. The learning rate is set to $1e^{-6}$, and the batch size is 4. To prepare the training set, the coded data without SAO were assumed as the network input. For convenient implementation, we applied the proposed network to 64 × 64 luminance CTUs only. However, its performance degradation is negligible.

C. PERFORMANCE EVALUATION

The full comparative results are shown in Table 1. First, the results show the proposed method outperforms all other schemes overall, especially for LDP mode with high margin.

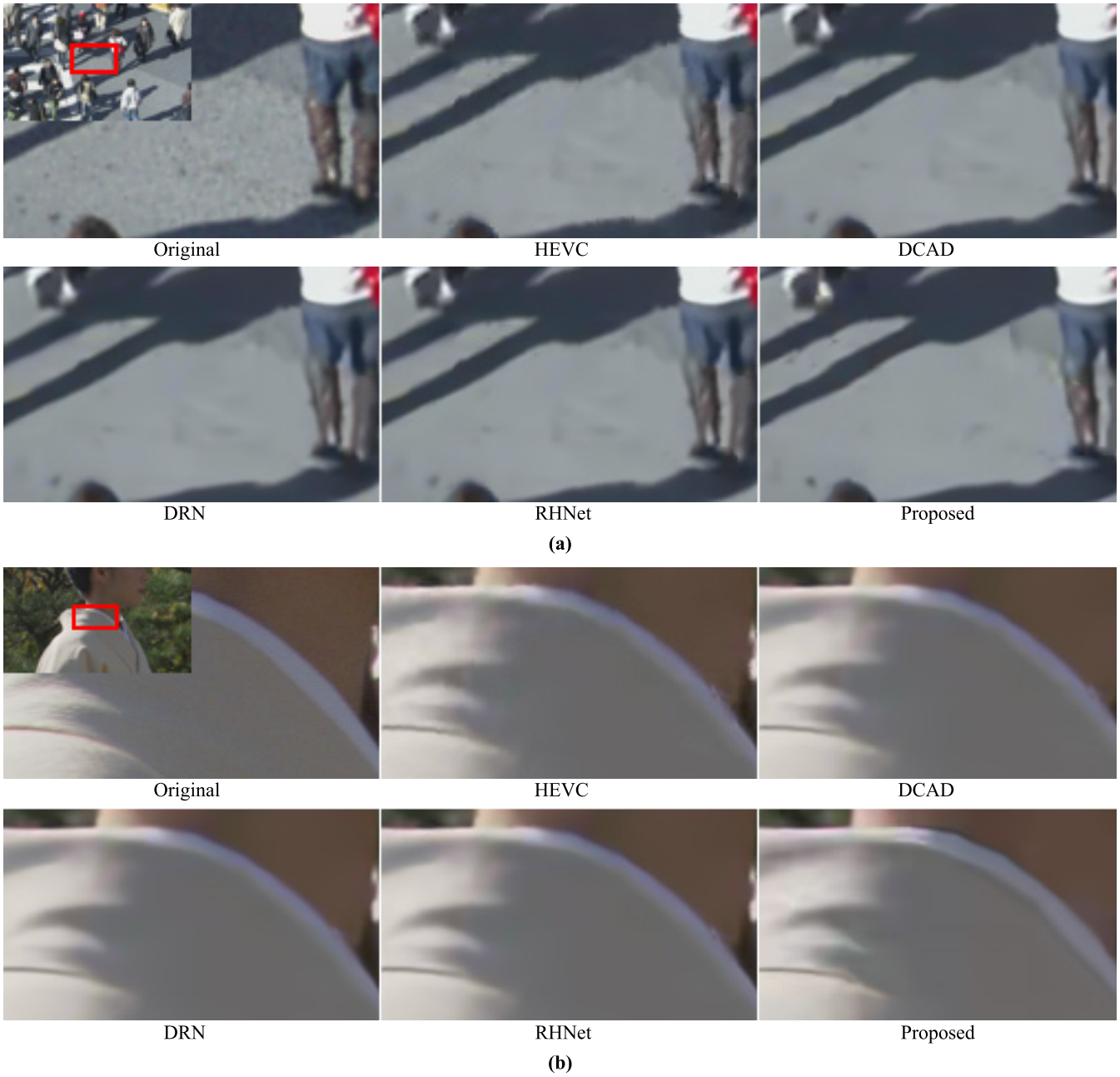

FIGURE 6. Visual quality comparison with the original HEVC in LDP mode: (a) PeopleOnStreet 7th frame (QP = 32), (b) Kimono 28th frame (QP = 37) for original, HEVC, DCAD, DRN, RHNet, and proposed method.

Other methods sometimes show the good performance, but they sometimes yield a negative gain. Likewise, the proposed scheme not only shows the best performance, but also stable performance for all configurations and sequences. This means the proposed method predicts well for both intra-coded and inter-coded distortions, and it is concluded that the highly complicated behaviors of error can be considerably well predicted by the offsets.

Another interesting point in the results is that the proposed scheme shows better performance for UHD and class A, B, E, i.e., high-resolution sequences, which have greater relevance. For better readability, the average performance for high- and low-resolution sequences is shown separately in Table 1. It is observed that all methods show greater performance improvement for high-resolution sequences. It is analyzed the network is trained by HD sequences, which matches Class B. Furthermore, it is found that the larger resolution contents are generally less complicated within a fixed 64×64 block. In other words, it would include fewer high-frequency components because of enlargement of the image. Therefore, it will be relatively easier to predict the types of error. It is worth noting that the performance inefficiencies for smaller resolution content are also observed in the conventional SAO. However, the results for UHD are worse than for Class A, which is because the training samples were prepared with Class B. If the training data can be prepared with UHD,

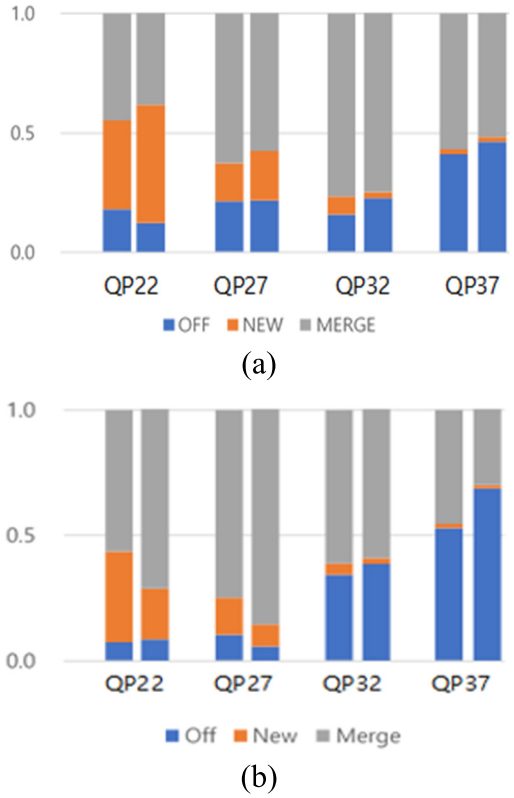

FIGURE 7. SAO mode ratio for (a) class A, (b) class B: The left bars for each QP value represent the ratio of HEVC, and right bars represent that of the proposed method.

then its performance will be much higher. Unfortunately, we cannot find suitable UHD contents for training, and it will remain as our future work.

For more analysis, the performance comparisons with R-D curves are shown in Fig. 5. It is obvious that the proposed method outperforms the HEVC anchor for all rate points, and it is also interesting to see a larger performance increase for a higher bit-rate. In addition to the R-D curves, visual quality comparisons are also provided in Fig. 6. We selected a frame where the proposed method used fewer bits than HEVC, but it achieves higher PSNR. The visual quality improvement is also cleared observed compared to other methods.

Finally, we analyzed how the proposed method changes the ratio of SAO mode by counting the numbers of CTUs applied SAO-New/Merge, or SAO-Off modes for the class A and B dataset in Fig. 7. In general, the ratio of SAO-On modes (New/Merge) reduces as QP increases for both the conventional and proposed methods. However, the ratio of SAO-New mode largely increases for lower QP in class A, which would give large performance improvement. It is analyzed that residual signal contains more information in lower QP, and the proposed deep network is capable to predict its complex pattern, while the conventional SAO in HEVC is not. On the contrary, the proposed network reduces the number of SAO-New modes for higher QP or lower resolution. The network was analyzed to have been trained to target highly complicated patterns, abandoning other patterns with small gains.

TABLE 2. Number of parameters, FLOPs, and average execution time comparisons for class B sequences.

Method	DCAD	DRN	RHNet	Prop.
#Params (K)	297	1109	4575	3578
#FLOPs (G)	615	2291	9488	6930
Runtime (ms)	110	501	1249	5670 (QP22) 2409 (QP27) 510 (QP32) 397 (QP37)

D. COMPLEXITY ANALYSIS

In this section, we compare the proposed method to other learning-based methods in terms of the model complexity. For a fair comparison, we executed all the algorithms on an NVIDIA GTX 2080Ti GPU. Table 2 lists the number of parameters and floating point operations (FLOPs), and average execution times for class B sequences. Because the proposed method is selectively applied for each CTU as the conventional SAO, the runtime highly depends on QP value. As can be seen from Table 2, the proposed method has fewer parameters and operations than RHNet, but more than DCAD and DRN. Therefore, the proposed method takes the second highest time on average as compared to other methods. Total execution time is greatly reduced for large QP value, because only about 1-2% of CTUs uses ECN during decoding.

V. CONCLUSION

In this paper, we propose a novel in-loop filter using deep learning with side information to replace SAO in HEVC. The proposed network designs are highly motivated by SAO-EO in HEVC, that is, one network classifies the types of error according to the edge shape of the reconstructed signal, and the other network simultaneously predicts the optimal offset values. Then, these offsets are transmitted to the decoder to strongly improve the estimation accuracy. Furthermore, we propose a modified compact syntax design. It showed a 4.2% bit-rate saving in LDP mode and 2.8% in RA mode on average, which outperforms other deep-learning-based in-loop or post-filter schemes.

REFERENCES

[1] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.
[2] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 9, pp. 1103–1120, Sep. 2007.
[3] V. Sze and M. Budagavi, "High throughput CABAC entropy coding in HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1778–1791, Dec. 2012.
[4] A. Norkin, G. Bjøntegaard, A. Fuldseth, M. Narroschke, M. Ikeda, K. Andersson, M. Zhou, and G. Auwera, "HEVC deblocking filter," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1746–1754, Dec. 2012.

[5] C.-M. Fu, E. Alshina, A. Alshin, Y.-W. Huang, C.-Y. Chen, C.-Y. Tsai, C.-W. Hsu, S.-M. Lei, J.-H. Park, and W.-J. Han, "Sample adaptive offset in the HEVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1755–1764, Dec. 2012.
[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[7] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.
[8] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 295–307, Feb. 2016.
[9] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1646–1654.
[10] K. Yu, C. Dong, C. Change Loy, and X. Tang, "Deep convolution networks for compression artifacts reduction," 2016, *arXiv:1608.02778*. [Online]. Available: http://arxiv.org/abs/1608.02778
[11] M. Rabbani and R. Joshi, "An overview of the JPEG 2000 still image compression standard," *Signal Process., Image Commun.*, vol. 17, no. 1, pp. 3–48, Jan. 2002.
[12] J. Guo and H. Chao, "Building dual-domain representations for compression artifacts reduction," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 628–644.
[13] W.-S. Park and M. Kim, "CNN-based in-loop filtering for coding efficiency improvement," in *Proc. IEEE 12th Image, Video, Multidimensional Signal Process. Workshop (IVMSP)*, Jul. 2016, pp. 1–5.
[14] Y. Dai, D. Liu, and F. Wu, "A convolutional neural network approach for post-processing in HEVC intra coding," in *Proc. Int. Conf. Multimedia Modeling*, 2017, pp. 28–39.
[15] R. Yang, M. Xu, and Z. Wang, "Decoder-side HEVC quality enhancement with scalable convolutional neural network," in *Proc. IEEE Int. Conf. Multimedia Expo*, Jul. 2017, pp. 817–822.
[16] C. Li, L. Song, R. Xie, and W. Zhang, "CNN based post-processing to improve HEVC," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, pp. 4577–4580.
[17] T. Wang, M. Chen, and H. Chao, "A novel deep learning-based method of improving coding efficiency from the decoder-end for HEVC," in *Proc. Data Compress. Conf. (DCC)*, Apr. 2017, pp. 410–419.
[18] C. Jia, S. Wang, X. Zhang, S. Wang, and S. Ma, "Spatial-temporal residue network based in-loop filter for video coding," in *Proc. IEEE Vis. Commun. Image Process.*, Dec. 2017, pp. 1–4.
[19] C. Jia, S. Wang, X. Zhang, S. Wang, J. Liu, S. Pu, and S. Ma, "Content-aware convolutional neural network for in-loop filtering in high efficiency video coding," *IEEE Trans. Image Process.*, vol. 28, no. 7, pp. 3343–3356, Jul. 2019.
[20] S. Kuanar, C. Conly, and K. R. Rao, "Deep learning based HEVC in-loop filtering for decoder quality enhancement," in *Proc. Picture Coding Symp. (PCS)*, Jun. 2018, pp. 164–168.
[21] Y. Zhang, T. Shen, X. Ji, Y. Zhang, R. Xiong, and Q. Dai, "Residual highway convolutional neural networks for in-loop filtering in HEVC," *IEEE Trans. Image Process.*, vol. 27, no. 8, pp. 3827–3841, Aug. 2018.
[22] Y. Wang, H. Zhu, Y. Li, Z. Chen, and S. Liu, "Dense residual convolutional neural network based in-loop filter for HEVC," in *Proc. IEEE Vis. Commun. Image Process.*, Dec. 2018, pp. 1–4.
[23] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.
[24] D. Wang, S. Xia, W. Yang, Y. Hu, and J. Liu, "Partition tree guided progressive rethinking network for in-loop filtering of HEVC," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 2671–2675.
[25] F. Bossen, *Common Test Conditions and Software Reference Configurations*, document JCTVC-L1100, 12, 2013.
[26] G. Bjøntegaard, *Calculation of Average PSNR Differences Between RD-Curves*, document VCEG-M33, 2001.
[27] *The Consumer Digital Video Library*. Accessed: Nov. 1, 2020. [Online]. Available: https://www.cdvl.org/
[28] *Xiph.org Video Test Media*. Accessed: Nov. 1, 2020. [Online]. Available: https://media.xiph.org/video/derf
[29] *Institut de Recherche en Communications et Cybernétique de Nantes*. Accessed: Nov. 1, 2020. [Online]. Available: http://ivc.univ-nantes.fr/en/databases/
[30] N. Ketkar, "Introduction to Pytorch," in *Deep Learning With Python*. Berkeley, CA, USA: Apress, 2017.
[31] *HM Software Version 16.9 and Software Manual*. Accessed: Nov. 1, 2020. [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-16.9/
[32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

SO YOON LEE received the B.S. and M.S. degrees from the School of Electronics and Information Engineering, Korea Aerospace University, Goyang, South Korea, in 2017 and 2019, respectively. Her research interest includes the image/video restoration and compression.

YOONMO YANG (Graduate Student Member, IEEE) received the B.S. and M.S. degrees from the School of Electronics and Information Engineering, Korea Aerospace University, Goyang, South Korea, in 2015 and 2017, respectively, where he is currently pursuing the Ph.D. degree. His research interest includes the image/video restoration and compression.

DONGSIN KIM received the B.S. degree from the School of Electronics and Information Engineering, Korea Aerospace University, Goyang, South Korea, in 2020, where he is currently pursuing the M.S. degree. His research interest includes the image/video restoration and compression.

SEUNGHYUN CHO received the B.S. degree in electrical engineering from Kyungpook National University, Daegu, South Korea, in 2003, and the M.S. and Ph.D. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2006 and 2015, respectively. He worked with the Electronics and Telecommunications Research Institute, Daejeon, as a Principal Researcher, from 2006 to 2019. He is currently an Assistant Professor with the Department of Information and Communication Engineering, Kyungnam University, South Korea. His research interests include video coding algorithms, HW/SW video codec design, and deep learning-based video signal processing.

BYUNG TAE OH (Member, IEEE) received the B.S. degree in electrical engineering from Yonsei University, Seoul, South Korea, in 2003, and the M.S. and Ph.D. degrees in electrical engineering from the University of Southern California (USC), Los Angeles, CA, USA, in 2007 and 2009, respectively.

From 2009 to 2013, he was a Research Staff with the Samsung Advanced Institute of Technology (SAIT), Samsung Electronics Company Ltd., South Korea. Since 2013, he has been with the School of Electronics and Information Engineering, Korea Aerospace University (KAU), where he is currently an Associate Professor. His research interests include image/video restoration and compression, and image/video forensics.