Contents lists available at ScienceDirect

Signal Processing: Image Communication

journal homepage: www.elsevier.com/locate/image

Gated fusion network for SAO filter and inter frame prediction in Versatile Video Coding

Shiba Kuanar [a],[*], Vassilis Athitsos [b], Dwarikanath Mahapatra [c], K.R. Rao [a]

[a] Electrical Engineering, University of Texas at Arlington, TX, USA
[b] Computer Science and Engineering, University of Texas at Arlington, TX, USA
[c] Inception Institute of AI, Abu Dhabi, United Arab Emirates

ARTICLE INFO

Keywords:
Artifacts
VVC
SAO
De-convolution
In-loop filter
Deep learning

ABSTRACT

In order to achieve higher coding efficiency, the Versatile Video Coding (VVC) standard includes several new components at the expense of an increase in decoder computational complexity. These technologies often create ringing and contouring effects on the reconstructed frames at a low bit rate and introduce blurring and distortion. To smooth those visual artifacts, the H.266/VVC framework supports four post-processing filter operations. The state-of-the-art CNN-based in-loop filters prefer to deploy multiple networks for various quantization parameters and frame resolutions, which increases training resources and subsequently becomes overhead at decoder frame reconstruction. This paper presents a single deep-learning-based model for sample adaptive off-set (SAO) non-linear filtering operation on the decoder side, uses feature correlation among adjacent frames, and substantiates the merits of intra–inter frame quality enhancement. We introduced a variable filter size dual multi-scale convolutional neural network (D-MSCNN) to attenuate the compression artifact and incorporated strided deconvolution to restore the high-frequency details on the distorted frame. Our model follows sequential training across all QP values and updates the model weights. Using data augmentation, weight fusion, and residual learning, we demonstrated that our model could be trained effectively by transferring the convolution prior feature indices to the decoder to produce a dense output map. The Objective measurements demonstrate that the proposed method outperforms the baseline VVC method in PSNR, MS-SSIM, and VMAF metrics and achieves an average of 5.16% bit rate saving on different test sequence categories.

1. Introduction

The number of digital videos is increasing dramatically due to the availability of high-resolution ultra-high definition (UHD) and virtual reality (VR) videos across social media and with wider dynamic ranges. The growth of these video contents consumes lots of network capacity and poses a challenge in storing and transmitting visual data over the channel. To address this demand, the Joint Video Exploration Team (JVET) of ITU-T Video Coding Experts Group (VCEG) [1] and ISO/IEC Moving Picture Experts Group (MPEG) [2] proposed the Versatile Video Coding (VVC), a new video coding standard [3] in July 2020, and adopted block-based hybrid coding schemes. Most compression algorithms, along with VVC, adopt a transform-based coding strategy, partition the video frames into slices, grids of tiles across rows and columns, apply quantization, motion estimation and motion compensation, context-based adaptive coding, and frame reconstruction. Compared with its predecessor H.265/HEVC [4], the upcoming video coding standard, H.266/VVC, achieves up to 40% BD-BR reduction with an increase in complex tree partitioning structure [5]. These high compression ratios often introduce undesired artifacts, blur the compressed images, and impact the visual quality. Despite several video and image compression techniques' success, an effective artifact reduction in-loop filter technique remains an active area of research and persists as an open issue for further investigations.

Most modern video compression techniques, including H.264/AVC [6], H.265/HEVC [4] uses block-by-block coding to check pixel heterogeneity in local regions and encode the video frame contents. Despite the merit of the block-based technique, the coding process leads to ringing, jittering, noise, and blocking artifacts in decoded images. The blocking artifacts in the compressed image mainly arise from three aspects [7]. Firstly, the difference of pixel values between blocks is magnified after quantization and inverse quantization. Secondly, in the motion estimation process, the selected reference blocks are sometimes copied from different positions of the neighboring reference frames. Thirdly, the blocking artifact can propagate from the previous frame to any current frame-block position during motion-compensation

[*] Corresponding author.
E-mail address: shiba.kuanar@mavs.uta.edu (S. Kuanar).

https://doi.org/10.1016/j.image.2022.116839
Received 19 January 2022; Received in revised form 6 July 2022; Accepted 9 August 2022
Available online 19 August 2022
0923-5965/© 2022 Elsevier B.V. All rights reserved.

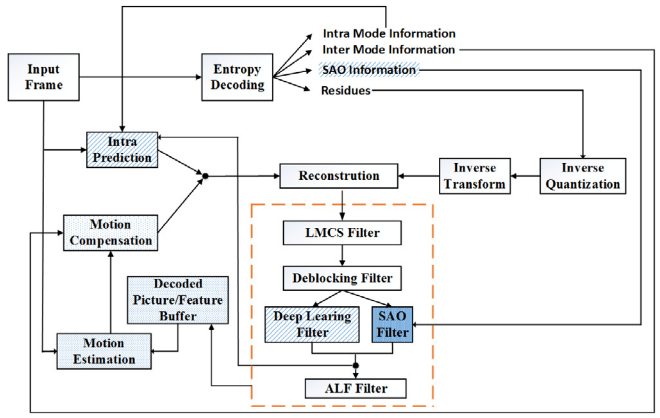


Fig. 1. VVC Decoder Block Diagram with SAO filter Replaced by our D-MSCNN Deep Learning Model. Diagonal stripe pattern rectangles covered with orange color dotted lines in the middle correspond to in-loop filtering coding blocks.

prediction [8]. These matching blocks are often not accurate and lead to pixel value discontinuity. Subsequently, the blocking artifact at the prediction block boundary weakens the input image's subjective quality. Additionally, the transformation of different frames follows an independent coding procedure with multiple parameters and leads to discontinuity in the residual signals. Besides that, the quantization distortion of the high-frequency co-efficient causes ripple phenomena around the sharp edges and introduces ringing artifacts [9]. These non-linear phenomena induce poor user experience and may not be adequate for real-time video applications.

To overcome the coding artifacts in compressed video, the H.266/VVC decoder framework comprises four low pass in-loop filters (Fig. 1). These built-in in-loop filters include a Luma Mapping with Chroma Scaling (LMCS), de-blocking filter, sample adaptive offset filter, and adaptive loop filter. A deblocking filter was developed to smooth the discontinuities based on mean pixel values at block boundaries and coded the block parameters [7]. Subsequently, the SAO filter was incorporated to access the inner block pixels and reduce distortion by adding offset to each sample. An offset can be a negative or positive integer and be added adaptively to each sample category. Presently, the SAO filter in VVC performs four 1-D edge pattern searches and then selects edge offset [10]. However, if the input block contains multiple edges, the four one-dimension edge offset approach may not be quite efficient to remove the artifacts from all directions. Therefore, combining diverse 1-D edge patterns to different edge types in a particular region can be an alternative approach. Subsequently, an adaptive loop filtering (ALF) method [2] was proposed to enhance the decoded quality of the current frame, store it in the buffer and predict the future frames. Similar to the theory of the Wiener filter, the ALF filter trains the filter coefficients and minimizes the mean square error between the buffered current frame and the reconstructed frame. Therefore the in-loop filters are stacked one above the other to overcome various distortions and enhance visual quality while saving coding bit-rate.

Besides the built-in SAO filters in VVC, a few alternative approaches are proposed progressively and primarily classified either as a learning or a heuristic-based method. In heuristic techniques [11,12], the statistical properties of the artifacts are primarily modeled according to the image priors and fused to the latter feature maps. These global features include gradient, contrast, signal strength, colors, texture heterogeneity, edges, and shape and vary within the frames [9]. Although the heuristic features boost the coding efficiency, these manually handcrafted feature extraction does not guarantee a good object descriptor and results in poor image quality reconstruction. Alternatively, it is intractable to build a multi-scale filtering framework and enhance the coding efficiency. Learning-based methods are introduced to address the above shortcomings, retain broader image features and optimize the feature space. On the other hand, the artifact removal methods can be extended barely from one compression technique to another. Subsequently, data-driven designs can be an alternative for better generalization of image reconstruction and quality enhancement in compressed frames.

We introduced a deep CNN model for artifact reduction in SAO output prediction. Our dual D-MSCNN model includes the feature extraction, enhancement, and mapping layers in succession and removes the undesired noisy features. The model learns the residual blocks between the neighboring frames and calculates the difference between the blocks in the current (target) frame to those in the reference (matching) frame. We observed that the effective feature learning improved by transferring the model parameters in shallow convolution modules to deconvolution modules [13] through skip connections and data augmentation. Finally, our model lowered the residual block artifacts evidently and retained edge patterns and sharp details.

Contribution: We acknowledge the deep learning-based in-loop SAO filtering as a design problem and a complete analysis is required to incorporate it in the decoder framework. The important contributions of our work are summarized as follows:

- We presented a gated fusion-guided framework in our model design, which effectively combines the inter–intra frame local and temporal feature heterogeneity. Our decoupled model includes a modified loss function to constrain the pixel errors and incorporates the intermediate convolution feature maps through skip connections. Collectively, our loss function can be viewed as a generalization of MSE at each batch and adds image gradients as priors for image reconstruction.
- A data-driven deconvolution framework is integrated into the decoder module to overcome the quantization artifacts. We collected a large-scale dataset for our SAO in-loop filter training. The end-to-end framework learns the feature map aggregation in separate sub-tasks, optimizes the parameters, and reduces the noise to a greater capacity.
- Our model's qualitative and quantitative evaluation shows the effectiveness of artifact removal, especially at crowded target regions, and performs favorably against the existing in-loop deep learning models.

We discuss the advances in video coding standards in Section 2 and describe the recent heuristic and learning-based approaches for VVC in-loop filters. We present a summary of the proposed framework design and discuss the key points of our method in Section 3. We elaborate on the experimental setting, evaluate results, and analyze its trade-offs in Section 4. Then, we draw the conclusions in Section 5.

2. Related work

In this section, we briefly discussed the prior work related to the H.266/VVC loop filtering approach. The current in-loop filter resides in the encoder and decoder loop, where the filtered images are stored in the picture buffer and used to predict the later coded blocks or reconstruct the whole frame. Apparently, they change the rate–distortion behaviors of each CU label, makes in-consistence in the rate optimization and finally CTU level model decision at different QP values. Eventually, a handful of in-loop filtering modules have been incorporated into the VVC architecture to improve coding efficiency in inter and intra predictions [14,15]. In addition to the H.266/VVC framework development, four major built-in in-loop filters are proposed to reduce the compression artifacts: (1) Luma Mapping with Chroma Scaling (LMCS) operation, (2) DBF filter, (3) SAO filter, and (4) ALF filter.

2.1. Luma Mapping with Chroma Scaling (LMCS)

The Luma Mapping with Chroma Scaling in-loop operation in H.266/VVC include two basic sequential operations: (1) Luma Mapping

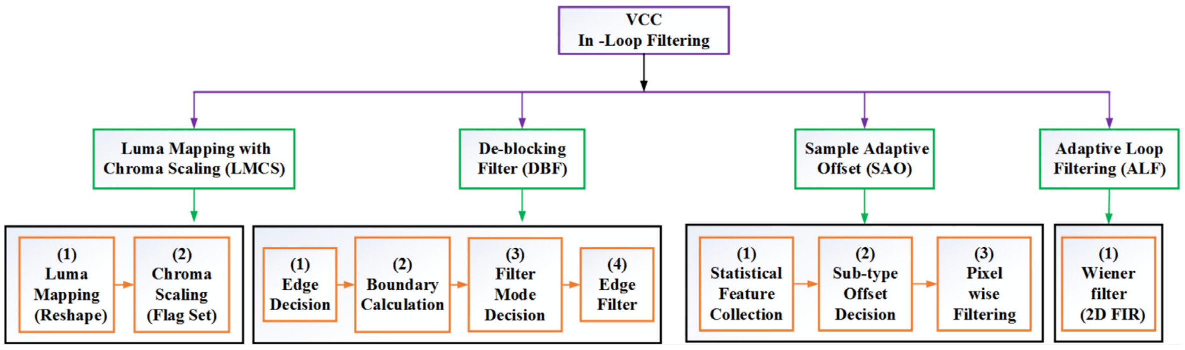

VCC
In -Loop Filtering
Luma Mapping with Chroma Scaling (LMCS)
De-blocking Filter (DBF)
Sample Adaptive Offset (SAO)
Adaptive Loop Filtering (ALF)
(1) Luma Mapping (Reshape)
(2) Chroma Scaling (Flag Set)
(1) Edge Decision
(2) Boundary Calculation
(3) Filter Mode Decision
(4) Edge Filter
(1) Statistical Feature Collection
(2) Sub-type Offset Decision
(3) Pixel wise Filtering
(1) Wiener filter (2D FIR)

Fig. 2. Primary in-loop filters in decoder side of versatile video coding and sub-processes for reconstruction.

and (2) Chroma Scaling [16]. Luma Mapping uses a adaptive piecewise linear reshaping function and scale the input luma code signals (amplitudes) across the dynamic ranges. The reshaping function takes the intensity lines from input, make it into sixteen segments on the curve, and finally boost up the signal precision. Therefore, the luma mapping makes better use of luma code range values allowed within specified bit depth and improves the coding efficiency for standard and high dynamic range video [17]. On top of Luma mapping, the Chroma scaling step applies some residual signaling on the Chroma components with flag activation. With the above two operations the decoder improves the signal, adopts signal characteristics, and thereby improves the coding efficiency. However, the LMCS filter does not categorically address the artifacts reduction and is therefore not included in our image enhancement analysis.

2.2. Deblocking Filter (DF)

The Versatile Video Coding standard introduces an advanced block partitioning with quadtree and nested multi-type tree structure and allows larger block sizes up to 128 × 128 samples compared to its predecessors, like H.265/HEVC [4] and H.264/AVC [6]. During this block-based quantization, transformation, and prediction, the discontinuities frequently exist at block boundaries and lead to blocking artifacts. Therefore the de-blocking low pass filter is included to remove the blocking artifacts at prediction units and smooths the edge discontinuities. As illustrated in Fig. 2, the de-blocking filter process in H.266/VVC is broadly divided into four stages, namely (1) edge decision, (2) boundary calculation, (3) filter mode decision, and (4) edge filtering [8]. To address these artifacts, the de-blocking filters in VVC follow a similar design to HEVC [4] and apply at 8 × 8 or 4 × 4 block boundaries [18]. After that, it calculates the boundary strengths according to the encoder information and selects possible values 0, 1, and 2, indicating no filtering, weak filtering, and robust filtering, respectively. If the boundary strength value is high, an additional condition assessment involves applying a weak or strong filtering mode based on boundary sample values [7]. After that, the edge filters are commonly employed horizontally and vertically for further smoothening.

2.3. Sample Adaptive Offset (SAO)

In 2012 C.-M. Fu et al. introduced the SAO [19] filtering technique in HEVC in-loop framework and subsequently integrated it along with the DBF filter in H.266/VVC. A sample adaptive offset filter aims to compensate the reconstructed samples by adding an offset to each pixel, correcting the local intensity changes, and attenuating the ringing and banding artifact induced by the coding process [7]. As indicated in Fig. 2, SAO filter operation includes three significant stages, specifically (1) statistical feature collection, (2) off-set decision and sub-typing, and (3) adaptive pixel wise filtering. In stage one, the number of pixels and their sum of distortions are calculated for a certain SAO type. In stage two, the method classify the reconstructed samples and select the offsets for each category. In current implementation, the SAO filtering operation broadly categorized into band offset (BO) and edge offset (EO) methods [2,9]. After that, the best SAO type and its corresponding offset are selected from the candidates of edge offsets and band offset types. Overall, the edge offset reduces the artifacts around all edge directions, and the band offset helps to reduce the artifacts in smooth regions with lesser complexity. Lastly, the CTU pixels are filtered in stage three by conditionally adding the corresponding best offset values. Consequently, the complexity of the SAO filter is much lower than other in-loop filters and makes it relevant for low-latency encoders.

2.4. Adaptive Loop Filtering (ALF)

During the H.266/VVC framework development, ALF [20] and cross-component adaptive loop filtering (CC-ALF) [21] were introduced to reduce potential distortion introduced during the quantization and transform process. Fig. 3 shows the ALF Luma and Chroma Filtering steps as defined in VVC specification. In ALF, each block is partitioned into CTU and each CTU is then divided into 4 × 4 macro blocks [16]. For the Luma component, ALU uses a 7 × 7 diamond share filter. For the Chroma components, the ALF uses a 5 × 5 diamond shape filter or a CC-ALF diamond filter. The proposed ALF is located at the last in-loop processing stage, applies to chroma and luma samples, and is regarded as a tool to capture and fix artifacts from previous stages. Based on local gradients, the ALF operation further minimizes the mean square error between all sample locations of reconstructed frames and the raw frames. ALF scans the decoded pictures at the CTU level and selectively applies several two-dimensional finite impulse response Wiener filters before the blocks become output for prediction. At the encoder side, the algorithm estimates the low-pass frequency filter coefficients that often lead to small errors, adaptively clips coefficient and then conveys the signals to the corresponding pixel coefficients at decoder side. As described in [16], the CC-ALF architecture within the ALF filter takes the co-located luma samples as input, independently generates the correction among the chroma channels, and eventually refines the chroma sample values.

2.5. Deep learning-based in-loop filter for VVC

Inspired by recent advances in deep neural networks, many CNN architectures have been proposed in different image enhancement or video compression tasks like intra–inter coding [22], motion compensated prediction, image reconstruction by in-loop filtering [23], edge detection [24], super-resolution [25], and contrast enhancement [26]. The proliferation of these models has been motivated by the fact that computer aided analysis might offer an automated, and sophisticated image reconstruction solution. In MIF-Net [23], the author proposed a single-frame block adaptive CNN model for the in-loop filter, constructed a diverse HIF dataset for model training, improved the CTU level artifact, and enhanced the visual quality for p-frame

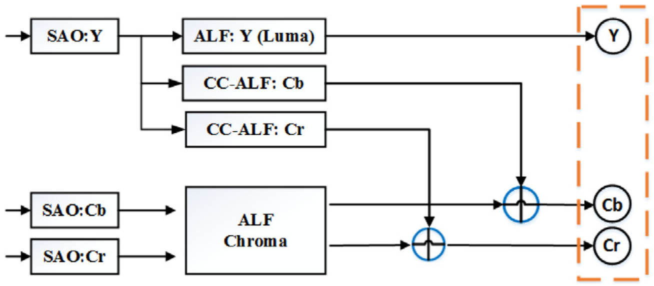


Fig. 3. ALF Luma and Chroma Filters along with CC-ALF in VVC.

HEVC profiles. S. Wang et al. [15] proposed an adaptive wavelet-based low pass filter with Haar transform and improved RD cost at CU label reconstruction in VVC intra coding. Their model improved the compression artifacts and coding efficiency by an average of 1.72% bit-rate reduction. Z. Huang et al. [27] proposed an agent-based adaptive deep reinforcement learning in VVC and included filtering output in the decision-making step. The model followed a two-stage training procedure and tuned the network parameters to perform better and lower the computational cost. D. Ma et al. proposed MFRNet [28] in-loop post-processing technique for multi-label feature learning, improved the feature information flow between blocks through dense cascaded connection, and finally improved the coding efficiency with consistent coding gain in bit-rates. M. Wang et al. [29] suggested an attention-based dual CNN network, considered the image quantization priors, luma, and chroma image partition information during feature fusion, and reduced artifact and noise effects in I and B frames.

2.6. Video quality assessment

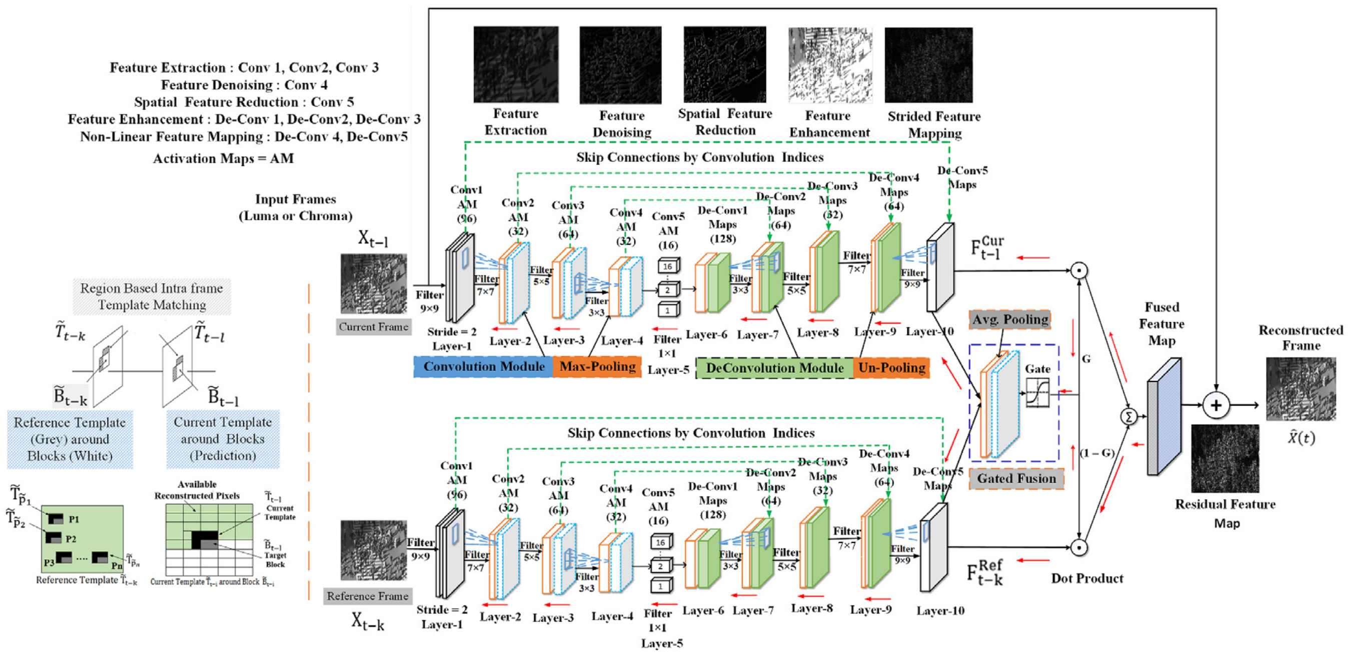
The video quality evaluations can be measured by both subjective and objective assessments. Over the last decades, many visual quality metrics have been proposed in H.265/HEVC and H.266/VVC encoder rate–distortion optimization and decoder filtering decisions to improve the coding efficiency [30]. The most widely used objective quality performance metrics are the peak signal-to-noise ratio (PSNR) [31], multi-scale structural similarity (MS-SSIM) [32,33] video multi-method assessment fusion (VMAF) [34,35] and used to measure the distortion between the source and reconstructed signal. The PSNR measure is not always consistent with the human visual system (HVS). The subjective quality assessment of a video [36] in HVS is measured by human judgments and evaluates the mean opinion score (MOS). These assessment process needs specialized domain expertise, and fell short of accurately capturing human perception, follow time consuming steps and sometimes do not provide the real quality assessments in higher resolutions video contents. Therefore, a better objective quality assessment is preferred in practical applications. The standard objective measurement is performed mathematically using pixel differences between compressed and reference frames and provides the measure of correctness. Another widely used objective metric is the SSIM which approximates the perceptual distortion based on structural information changes across multi-scale frames. More recently, Netflix introduced machine learning-based VMAF technique which fuses several existing spatial and temporal objective metrics and estimates perceptual quality scores by the SVM regression technique. Multiple recent work [5,37], and [28] have demonstrated that VMAF has higher accuracy than conventional metrics and subjective assessments.

3. Proposed CNN framework

The deep learning community has progressed in various computer vision tasks with the ability to automatically extract features from multi-scale architectures and improved evaluation performance in recent years. Conversely, our image enhancement network uses a dual network with symmetric convolution and de-convolution operation and operates on different frame types. We proposed a CNN-based decoupled network for the artifact reduction in H.266/VVC intra–inter frame coding. The decoder block diagram of our D-MSCNN architecture is shown in Fig. 4 and thereafter integrated into the decoder side of the VVC (Fig. 1), VTM 8.0 framework [38]. The residue image is input into in-loop network, and the encoded features are learned between the current and predicted frames. The end-to-end paired architecture incorporates a series of ten convolutional and symmetric deconvolution streams and operates jointly through skip connections [9,39]. The convolutional filters act as a feature extractor, learn the spatial features, set out different intermediate activation maps, and down sample the input image contents into smaller abstractions through polling operations. Then subsequent un-pooling and deconvolution operations bring back the abstractions to the original resolution. Our deconvolution module consists of a hierarchy of up-convolutions, and each one corresponds to the respective convolutions indexes. The features extracted on the first convolution layer are pretty noisy. Consequently, we followed three immediate steps: feature denoising, contraction, and enhancement. The final mapping layer employs strided convolution, maps attributes to another intermediate vector, and restores the higher-level features. Finally, the residual layer assembles the discriminative patch representation, fuses their feature maps, and adds those to the input frame to generate the reconstructed frame $\hat{X}(t)$ (Fig. 4). The residual feature maps for the upper and lower network are defined as below:

$$\begin{gathered}F_0(Y) = Y_0 \\ F_i(Y) = max(0, w_i * Y_i + b_i), where\, i \in \{1 to 10\} \\ F^{Cur}_{t-l} = \overbrace{w^{Cur}_{10} * F_9(Y_i) + b_{10}}^{F^{Cur}_{10}(Y)} + X_{t-l} \\ F^{Ref}_{t-k} = \overbrace{w^{Ref}_{10} * F_9(Y_i) + b_{10}}^{F^{Ref}_{10}(Y)} + X_{t-k}\end{gathered} \tag{1}$$

Let X = $\{X_1, \dots, X_{t-k}, ..X_{t-l}, X_t, ..\}$ denote the input video sequence, where X_t is the frame at time step t, X_{t-l}, X_{t-k} current and reference frames respectively. In Eq. (1), the F_i, w_i, and b_i represent the feature map function, weight, and bias components in the i_{th} layer, and the symbol $*$ designate the convolution operation. The F^{Cur}_{t-l} and F^{Ref}_{t-k} denote the features maps of current and reference frame at the final deconvolution layer (Fig. 4). Each layered framework in the dual network is denoted as $n_1(f_1)$-$n_2(f_2)$-$n_3(f_3)$-$n_4(f_4)$-$n_5[f_5]$-$s[f_6]$-$s[f_7]$-$s[f_8]$-$s[f_9]$ -$s[f_{10}]$, where f, [ ], n, s represents the filter size, deconvolution filters, number of convolution filters, and stride respectively. Based on the network settings in Table 1, each network is represented as 96(9)-32(7)-64(5)-32(3)-16(1)-128[3]-64[5]-32[7]-64[9]-1[9]. Filter size of 3 × 3 selected for nonlinear mapping and spatial shrinking operations to reduce comprehensive computational complexity. The features extracted from the first three layers are comparatively noisy. Therefore, we introduced 2 × 2 max-pooling layers and reduced the feature dimensions after each convolution layer. In the convolution module (left, Fig. 4), we used three max-pooling operations to reduce the spatial size and control the model over-fitting. Although the pooling layer lowers the number of parameters, specific spatial localization information is lost within the receptive field. Therefore, three symmetric un-pooling layers are employed in the deconvolution module (right, Fig. 3) to perform the reverse pooling operation, enlarge the activation map, and reconstruct the original input size. The output activation map of the un-pooling layer is sparse. Consequently, the up-convolution operations are incorporated at each layer to enlarge the receptive field and retrieve more contextual information at a lower computation. Subsequently, the deconvolution layers densify these sparse feature maps through several convolution low-pass filter operations. In this way, the deconvolutional layer associate the input feature maps with



Feature Extraction : Conv 1, Conv2, Conv 3
Feature Denoising : Conv 4
Spatial Feature Reduction : Conv 5
Feature Enhancement : De-Conv 1, De-Conv2, De-Conv 3
Non-Linear Feature Mapping : De-Conv 4, De-Conv5
Activation Maps = AM
Feature Extraction
Feature Denoising
Spatial Feature Reduction
Feature Enhancement
Strided Feature Mapping
Skip Connections by Convolution Indices
Input Frames (Luma or Chroma)
Current Frame
Reference Frame
Region Based Intra frame Template Matching
Reference Template (Grey) around Blocks (White)
Current Template around Blocks (Prediction)
Available Reconstructed Pixels
Reference Template
Convolution Module
Max-Pooling
DeConvolution Module
Un-Pooling
Avg. Pooling
Gate
Gated Fusion
Fused Feature Map
Residual Feature Map
Reconstructed Frame
Dot Product
Layer-1
Layer-2
Layer-3
Layer-4
Layer-5
Layer-6
Layer-7
Layer-8
Layer-9
Layer-10
Stride = 2
G
(1 – G)

Fig. 4. Overall architecture diagram of our deep learning model for intra or inter frame reconstruction. The upper and lower dual network are parallel to one another and each contains symmetric layers of convolution, deconvolution, and includes: Feature Extraction, Denoising, Feature Reduction, Enhancement, Non-Linear Mapping and Reconstruction Layer. We merged the current and reference deconvolution feature maps to a gate array and weighted the contribution of each block for final image reconstruction.

Table 1
Layer wise configuration summary of MSCNN network (upper and lower) and their percentage parameter contribution.

Layers: (ℓ)	$\ell 1$	$\ell 2$	$\ell 3$	$\ell 4$	$\ell 5$	$\ell 6$	$\ell 7$	$\ell 8$	$\ell 9$	$\ell 10$
Convolution, stride = 1	Conv1	Conv2	Conv3	Conv4	Conv5	De-Conv1	De-Conv2	De-Conv3	De-Conv4	De-Conv5
Filter size ($j \times j$)	f_1: 9 × 9	f_2: 7 × 7	f_3: 5 × 5	f_4: 3 × 3	f_5: 1 × 1	f_6: 3 × 3	f_7: 5 × 5	f_8: 7 × 7	f_9: 9 × 9	f_{10}: 1 × 1
Activation maps (n)	n_1: 96	n_2: 32	n_3: 64	n_4: 32	n_5: 16	n_6: 128	n_7: 64	n_8: 32	n_9: 64	n_{10}: 1
Parameters	7872	150 560	51 264	18 464	528	18 560	204 864	100 384	165 952	65
Total parameters = 718 513										
Percentage (%)	1.1	20.95	7.13	2.57	0.07	2.58	28.51	13.97	23.1	0.01

multiple output noise locations often ignored in previous convolutional layers (left, Fig. 4). Our model experimented on the input frames' luminance and chrominance color components. The model follows a sequential training procedure, uses one luma and two chroma CTU samples as input, and shares the weight during the model training. The ReLU non-linear activation function is applied to the filter output responses. The dual CNN model parallelly learns the residue features maps $F_{10}(Y)$ for each of the current and reference frames. Consequently, the input image is combined with the residual to form the reconstructed image at the end.

3.1. Intra frame template matching prediction

Our model extends the prediction of intra-frame block matching task and uses the neighborhood-based template search. The template matching exploits the correlation or spatial redundancies between the pixels in target blocks and the reconstructed pixels from close surrounding blocks inside the same frame and minimizes the matching error. The location of these blocks is described by respective displacement vectors and restricted to certain patch regions. As described in Fig. 3, let the block $\tilde{B}_{t-i}$ represents the present target block on a current frame X_{t-l}, and the $\tilde{T}_{t-i}$ represents a template formed by a group of pixels overlapping on top and to the left of the target block $\tilde{B}_{t-i}$. As shown in Fig. 3, the candidate blocks may include multiple blocks P_1 to P_n in reference template $\tilde{T}_{t-k}$, leading to $\tilde{T}_{\tilde{P}_1} \ldots \tilde{T}_{\tilde{P}_n}$ matched template in the field of view. Thus the best-matched template $\tilde{T}_{\tilde{P}}$ was searched within the reconstructed (reference) template $\tilde{T}_{t-k}$ by minimizing the MSE between $\tilde{T}_{t-i}$ and templates between $\{\tilde{T}_{\tilde{P}_1}$ to $\tilde{T}_{\tilde{P}_n}\}$.

3.2. Gated feature fusion

The gated fusion layer is introduced to effectively combine the multi-frame local and temporal features from previous frames for current frame reconstruction. As illustrated in Fig. 3, the proposed fusion operation is composed of three layers: average pooling, convolution, and gated sigmoid layer. The $F^{Cur}_{t-l} \in R^{c\times h\times w}$ and $F^{Ref}_{t-k} \in R^{c\times h\times w}$ denote the probability maps current and reference feature space at the final deconvolution layer (Fig. 4). The symbol 'c' denote the number of feature channels or filter length, 'h' height, and 'w' the width of the map. The feature maps F^{Cur} and F^{Ref} from Eq. (1) are concatenated to obtain a fused probability map $F^{fusion}_{k-l} \in R^{2c\times h\times w}$. Thereafter, we employed global average pooling and sequentially, a spatial convolution operation with weights $W \in R^{c\times 2c\times 1\times 1}$ and c filters with a dimension of $2c\times 1\times 1$ per filter. The weights are learned to correlate the two feature maps from different frame regions and average their contributions for the prediction of forthcoming frame-blocks. Therefore, the output of the convolution layer is a coefficient matrix $G \in R^{c\times h\times w}$ and represented as:

$$G_{k,i,j} = \sum_{k'=1}^{2c} F^{fusion}_{k',i,j} \times W_{k',k,i,j} \tag{2}$$

$\forall k \in [1, c], i \in [1, h], j \in [1, w]$.

Finally, a sigmoid squashing function was applied to regularize G and map the $G_{k,i,j}$ values in the range $\in [0, 1]$. We term the $G^{Cur} = G$ and $G^{Ref} = (1 - G)$ as the weighted gates, where $G^{Cur}_{k,i,j}$ and $G^{Ref}_{k,i,j}$ denote how confidently one can rely on current and reference feature maps respectively to predict the pixel (i, j) in channel k [40]. The two coefficient matrices are then utilized to weigh the contributions of current and reference as follows:

$$\tilde{F}^{Cur} = F^{Cur} \odot G^{Cur}$$
$$\tilde{F}^{Ref} = F^{Ref} \odot G^{Ref} \quad (3)$$

where $\odot$ denote Hadamard dot product. Finally, we generated the gated fusion probability feature map as a weighted combination of F^{Cur} and F^{Ref}.

$$\tilde{F}^{fusion} = \tilde{F}^{Cur} + \tilde{F}^{Ref} \quad (4)$$

We predicted the feature residue map by a $\tilde{F}()$ fusion function and leverage the current frame map to optimize the total network via stochastic gradient descent.

3.3. Loss function

We investigated diverse model settings and model depth through a sequence of experiments and evaluated the in-loop filter performance. Considering the H.266/VVC framework's complexity, a CNN model with depth ten was introduced into our SAO filter analysis. The pixel intensity differences were used to measure the image similarity and formulated our study as a regression problem. We computed the mean squared error from a given batch of compressed ground truth residual images $(x_i)^n \in \{X_i\}$ and their corresponding feature maps. The Y_i represents the intermediate predicted features (texture, edge, and shape) learned along with the model layers. To constrain the reconstruction error, we set the training loss as a combination of pixel error (MSE) and convolution feature maps with a hyperparameter λ. The regularization parameter λ is empirically calculated to balance the reconstruction error and the sparse feature maps (Y_i^{ℓ}) learned from prior activation maps. The total loss function is expressed as below:

$$L(\theta) = \frac{1}{2N}\sum_{i=1}^{n} \overbrace{\left\|X_i - \hat{X}_i\right\|^2}^{\text{Direct Error Term}} + \lambda \sum_{\ell=1}^{5} \overbrace{|F_i^{\ell}(Y_i, \theta_i)|_1}^{\text{Convolution Prior features}} \quad (5)$$

$F_i^{\ell}(Y_i, \theta_i) = \left\{F_{Conv}^{Cur} + F_{Conv}^{Ref}\right\} \oplus f_{j\times j}$ contributed by current and reference frame Convolution prior features as stated in Eq. (1)
$\theta_i = model\ parameters\ (weight)\ at\ each\ image\ batch$
$f_{j\times j} = derivative\ filters\ with\ kernel\ size\ (j \times j)$
$\ell = convolution\ layers\ starting\ from\ 1\ to\ 5$
$Y_i = Activation\ map\ at\ layer\ \ell\ and\ sample\ x_i$
$n = number\ of\ training\ samples\ in\ each\ batch$
where θ_i represents the network parameter and includes both weight w_1 to w_{10} and bias terms b_1 to b_{10}, and n is the number of samples in a training batch. Our goal is to learn the loss function that maps the predicted residue values from input frames. The mapping function $F_i()$ performs the intermediate in-loop filtering and optimizes the θ parameters through iterative training steps. The parameters were optimized by minimizing the L_2 loss between X_i's ground truth blocks, and corresponding residual block map $F_{10}(Y_i, \theta)$. The reconstructed residual feature maps are then added to the input image to retrieve the final reconstructed frame $\hat{X}_i$. On the other hand, the MSE loss may generate blurry images and potentially result in the loss of fine details in the input image. Therefore, we incorporated an auxiliary loss term F_i^{ℓ} defined by the intermediate feature space learned from prior convolution layers.

In Algorithm 1, $x_i \in X_i, \mathbb{R}^{N\times N}$ denote the ith training sample out of n sample patches in a batch. The model predicts the output patch $\hat{x}_i \in Y_i$ and tries to approximate the input patch x_i. Our training scheme follows an iterative forward and backward propagation strategy and concurrently optimizes the model parameters. The network weight, bias, and hyper-parameter values are first initialized in Algorithm 1 and followed for N_{iter} iterations. A batch of image patches is processed by several steps and merged at the final layer for the whole image restoration. To make the reconstructed output image $\hat{X}_i$ close to X_i,

Algorithm 1: Model Optimization Steps

```
Require: Model parameter initialization $\theta$: $(W_0, B_0)$, learning rate
         $(\eta)$, regularization param: $\lambda \geq 0$
Input  : No. of iterations $N_{iter}$, epochs $N_{epoch}$. A batch of $(x_i)^n$
         sampled from training set $\{X_i\}$, patch size of $128 \times 128$
. Output : Trained CNN with mapped $\{\hat{X}\}$, $F^{\ell}(Y)$

foreach no_epoch in $N_{epoch}$ do
    foreach $k$ in $N_{iter}$ do
        Sample batch of 32 patches $\{x_i\}^n$
        foreach $i$ in $n$ do
            $\hat{x}_i \leftarrow x_i + F_i^{\ell}(\theta_i, Y_i)$            in Eq. (1)
            $\nabla L_k^{(i)}(\theta_i, \lambda) = \nabla F_i(\hat{x}_i, x_i, \nabla x_i, \nabla\theta_i) +$
                $2\lambda_1 \nabla \hat{F}_i^{\ell}(\nabla x_i, \nabla\theta_i) + \frac{\lambda}{2n}\left\|w_i\right\|^2$
                                                    in Eq. (2)
        end
        $\hat{X}_i = \sum_{i=1}^{n} \hat{x}_i$
    end
    Update $w_k \leftarrow w_k - \eta\frac{\partial}{\partial w_k}L(F_i(x_i, Y_i, \theta_k(w_k)) + 2\lambda w_k$
end
```

our training followed a mini-batch stochastic gradient descent and performed L2 norm optimization. In Eq. (5), The regularization component brought down weight, bias values, and $F_i^{\ell}(Y_i)$ matrix and effectively reduced the over-fitting.

4. Experimental results

We integrated our residue learning model into the VTM 8.0 [38] H.266/VVC framework to validate the model's performance. Our proposed D-MSCNN model is compared with other latest in-loop and SAO filtering methods and follows the standard test conditions defined in [41]. To verify our algorithm performance, we validated the video test sequences of classes A, B, C (WVGA), and D (WQVGA). We initially focused on intra coding model performance with AI mode configuration in VTM 8.0 software package and extended it to other inter modes. The input video frame sequences are encoded at various QP values 22, 27, 32, and 37. The PSNR, MS-SSIM, VMAF objective measurements are calculated for the performance evaluation, and the findings are shown in Tables 3 and 4.

4.1. Data pre-processing and training setup

We collected a large-scale dataset to provide sufficient diverse data for our model training and expedite the subsequent tasks. For our model training, we derived 750 high definition (HD) high resolution images from DIV2K dataset [42], 85 sequences from the Consumer Digital Video Library [43] in the Video Quality Experts Group (VQEG) [44], and obtained 92 supplementary sequences from BVI-DVC [45]. Note that a set of 52 sequences with five different resolutions are randomly selected from Xiph.org [46] and kept aside for model testing. Apart from that, we also kept eighteen naïve test video sequence of classes A to E from JCT-VC test set [47] and used those for evaluating our final model performance. The training and test video frames had no overlap and were kept in respective folders to demonstrate the generalizability of model validation. To reduce the correlation between the channels, color images (chroma and luma) are decomposed into YUV 4:2:0 file format and encoded by VTM-8.0 at different QP values. All raw videos are encoded under All-intra, Random Access, Low Delay P picture (LDP) and Low Delay B picture (LDB) configurations. The built-in loop filters are all enabled while compressing the video sequences. The Y luminance channel of video sequences contains the most visual information and proceeds as input to the network during the training phase. For training the CNN model, the pixel values of input video frames are standardized and normalized between 0 to 1 from their original intensity range. To augment the training data, images are rotated by an angle of 45° and 135°, scaled by a factor of 0.2, 0.6, and 0.8, and flipped vertically.

Our D-MSCNN network implementation is derived on an open-source Python/C++ based Tensorflow Keras deep learning library. All the experiments are carried out on a desktop I7 computer and NVIDIA GTX 1080 Ti GPU@1582 MHz graphic card with 11 GB RAM memory. Each image in the training set is decomposed into n number of overlapping patches and put into a set $X = \{X_i\}$, $i \in (1 \ldots n)$. The patch size 128×128 is chosen judiciously to accommodate the maximum encoder CTU structure and integrated to get a frame-wise reference after deconvolution. The model uses three CTUs 128×128 as input, including one luma and two chroma Cb, Cr samples, and shares the wight during the model training. Each input image was compressed by inter–intra coding at four distinct QP values {22, 27, 32, 37}. The trained network is applied to each patch X_i in an input image and subsequently constructs the de-noise output image by combining the results from all output $Y = \{Y_i\}$, $i \in (1 \ldots n)$ patches in a simple manner. Overall, the augmented $1800 \times 24 = 43\,200$ images are converted into 8696k training patches. The network is trained separately at each QP values and updated hyper-parameters for fine-tuning. The $L(\theta)$ loss metric was determined by comparing the ground-truth X_i patches and corresponding up-convoluted outcomes. The weight matrices in each layer are altered in terms of gradients and learning rates and propagated loss back into the neural network. During the training, we used a smaller learning rate (η) of 0.5×10^{-4} in the last layers and 0.5×10^{-3} in the remaining layers for better convergence. The learning rate is divided by 100 when the loss value becomes stable. The model performance was assessed by applying repeated stratified five-fold cross-validations. The performance was measured in all the validation folds and averaged across all epochs to generate stable outcome results. The trained files were compiled, collected for subsequent QP parameters, and integrated into VTM software [38]. We then turned off the base SAO filter from the configuration file for our model testing, performed an end-to-end operation, and after that compared the outcome with base SAO in-loop filter performance.

4.2. Random access configuration and QP mapping

The RA configuration parameters are updated in the `cfg/encoder_randomaccess_vtm_gop32.cfg` [48] configuration file of VTM-8.0. The QP value, frame rate, and spatial resolution settings are included in the sequence configuration file and used to perform the rate–distortion evaluation. The goal is to choose a group of pictures that provides the highest quality and utilizes the lowest amount of bandwidth. We set an intra-key frame period of one second, i.e. a set of 30 sequential frames between two I-frames. To overcome the poorer video quality longer group of pictures (GOP) of size 32 was selected for stationary sequences, and a short GOP of size 16 was selected for sequences with lots of dynamic content like high action or impact. To optimize perceived image quality, we attempt to adopt different GOP labels according to the specified target label and buffer occupancy. The test points are obtained for each scene using different quantization parameter values to cover a wide range of target bit rates. A larger QP value indicates a larger quantization step leading to a small bit rate. Consequently, these larger QP values introduce significant distortion and impact the frame quality reconstruction. For better temporal and local texture prediction, we used lower QP values in the hierarchical GOP structure in RA and low delay coding configuration. A higher QP value is assigned to the pictures that are less frequently referred to by neighboring pictures in GOP [49]. Therefore, different QP values are determined based on GOP labels between I, P, and B predictions and reordered pictures within the GOP period to achieve better compression.

Table 2
Coding time complexity comparisons of various in-loop SAO filters in AI mode and GPU configuration.

Class (Resolution)	t_{VTM} (frame/Sec)	(%) Decoding time overhead: $\Delta t/t_{VTM}$		
		ARLF [27]	MFRNet [28]	Our model
Class A (2560 × 1600)	3.763	35.2	28.4	8.1
Class B (1920 × 1080)	2.475	48.4	34.5	11.4
Class C (832 × 480)	1.514	53.3	41.7	15.2
Class D (416 × 240)	0.964	61.9	50.2	18.9
Class E (1280 × 720)	2.028	51.8	38.3	10.5
Average	–	50.1	38.6	12.8

Most hybrid video codecs, including VVC, use scalar uniform reconstruction quantizers to quantize the transform coefficients. The quantization step follows a nonlinear and irreversible step that takes the input values and maps them to integer quantization indexes. These indexes are transmitted using entropy coding at the encoder side, and then the decoder maps these quantization indexes to reconstruct the samples. The quantization step aims to decrease the bit rate required for sending the quantization indexes while maintaining a low reconstruction error and controlling the distortion label. A higher or coarser QP value indicates a larger quantization step, throwing the high-frequency coefficients and leading to higher distortion. The QP value in VVC is limited to values between 0 and 63, where zero is the best quality and 63 is the best compression. As stated in [50], the normalized QP-map is calculated for each frame and standardized by the maximum QP value, and 63 in VVC. We applied the cut-off threshold values and divided the entire QP range into four to five QP bands. We compressed the training sequence and trained the model parameters for each QP value. First, we selected the lowest QP value for our model training. Subsequently, those model weights were used to initialize the model weights for higher QP values and progressive trained all the QP values for better generalization.

4.3. Result comparison with CNN models

In this section, we provided the experimental results to demonstrate the effectiveness of our proposed method and compared it with H.266/VVC Baseline [38] and recently proposed ARLF [27] and MFR-Net [28] learning-based methods. We observed that our network outperformed other models based on various evaluation metrics and reconstructed the input image with less compressed artifacts. Additionally, the computational complexity of different models is compared in terms of decoding time at different test sequences and resolutions (Table 2).

4.4. Time complexity analysis

The computational complexity is one of the essential metrics for practical video compression systems. This section assesses our model's coding time complexity overhead in GPU mode under RA configuration and compares it with other deep learning in-loop methods. The frame inference time to decode a single frame inference in VTM baseline is provided in Table 2 and denoted as t_{VTM} and primarily depends on bit-rate, in-loop configuration, system memory transfer and sequence content [51]. The decoding time overhead for each reference model is denoted as t_{Ref}. Based on t_{VTM}, we measured the time overhead $\Delta t = (t_{Ref} - t_{VTM})$ introduced by each deep learning in-loop filter model and reported the ratio $\Delta t/t_{VTM}$ in the rest of columns in Table 2. The decoding time translates directly to the decoding cost and proportionate to the frame reconstruction. Therefore, a larger ratio indicates a relatively longer time overhead of the in-loop filter operation. Table 2 summarizes the average run time of representative learning-based CNN methods across JCT-VC (JVET) test sequences [47] at different image resolutions. All experimenters are implemented in a Keras machine

Table 3
RD performance comparison, BD-PSNR (dB) gain of our D-MSCNN model and Other state-of-the-art methods on the JCT-VC test-set at different resolutions and configurations.

Class (Resolution)	Sequence	Configuration											
		AI						RA					
		ARLF [27] vs. VTM		MFRNet [28] vs. VTM		Our model vs. VTM		ARLF [27] vs. VTM		MFRNet [28] vs. VTM		Our model vs. VTM	
		BDPSNR	BD-BR(%)	BDPSNR	BD-BR(%)	BDPSNR	BD-BR(%)	BDPSNR	BD-BR(%)	BDPSNR	BD-BR(%)	BDPSNR	BD-BR(%)
Class A1 (3840 × 2160)	Campfire	1.352	−1.091	1.857	−4.032	1.024	−5.044	0.94	−2.425	1.148	−4.291	1.014	−5.106
	FoodMarket4	1.041	−2.164	1.652	−3.102	1.227	−5.612	1.013	−3.12	0.745	−5.118	1.115	−5.452
	Tango2	0.706	−1.871	1.551	−2.405	1.134	−4.255	0.87	−2.47	1.046	−4.036	1.083	−5.241
Class A2 (3840 × 2160)	ParkRunning3	0.961	−2.621	1.315	−3.508	1.038	−5.42	1.032	−2.636	0.981	−3.447	1.072	−6.455
	CatRobot1	1.237	−2.865	1.104	−3.675	1.015	−5.563	0.857	−3.285	0.954	−4.394	1.103	−4.602
	DaylightRoad	1.154	−2.272	1.524	−2.863	1.372	−4.163	0.817	−2.628	1.172	−4.271	1.051	−5.231
Class B (1920 × 1080)	MarketPlace	0.916	−3.29	1.131	−3.881	1.173	−4.304	0.968	−2.52	1.085	−4.28	1.039	−4.513
	BasketballDrive	0.817	−1.91	0.853	−5.22	1.183	−5.37	0.854	−2.81	0.709	−4.85	1.017	−5.357
	BQTerrace	0.636	−2.75	0.638	−4.48	1.082	−4.957	0.792	−1.391	1.174	−4.152	1.088	−5.345
	Cactus	0.541	−1.153	0.526	−4.834	0.868	−3.914	0.504	−1.821	0.767	−4.724	1.091	−6.586
	RitualDance	0.415	−1.48	0.474	−3.74	1.021	−4.58	0.643	−2.19	0.813	−4.671	1.075	−5.142
Class C (832 × 480)	PartyScene	0.292	−2.279	0.349	−2.912	0.913	−6.034	0.909	−1.575	1.015	−3.902	1.018	−5.028
	BasketballDrill	0.406	−1.601	0.383	−3.754	1.062	−5.808	0.732	−1.164	0.898	−4.247	1.089	−6.053
	RaceHorses	0.562	−2.628	0.451	−4.672	1.019	−4.531	0.564	−1.592	0.824	−3.191	1.204	−4.372
	BQMall	0.551	−2.862	0.435	−5.747	0.532	−5.024	0.494	−1.903	0.706	−3.75	1.052	−5.192
Class D (416 × 240)	BlowingBubbles	0.842	−2.815	0.823	−4.714	0.463	−6.14	0.682	−2.235	0.689	−3.837	1.014	−4.572
	BQSquare	0.761	−2.27	0.753	−5.072	1.34	−5.851	0.473	−1.745	0.716	−4.303	1.134	−6.704
	BasketballPass	0.388	−1.34	0.409	−4.528	1.076	−5.904	0.441	−2.605	1.073	−4.65	1.059	−5.284
Class E (1280 × 720)	Johnny	0.194	−2.652	0.516	−3.611	0.943	−3.571	0.763	−3.586	0.935	−3.796	0.938	−4.825
	KristenAndSara	0.125	−2.284	0.245	−4.786	1.109	−4.46	0.907	−3.01	1.014	−4.083	1.043	−5.302
	FourPeople	0.374	−3.38	0.705	−3.905	1.268	−5.063	0.217	−4.136	0.965	−4.472	1.016	−5.117
Overall		0.679	−2.248	0.842	−4.068	1.041	−5.027	0.736	−2.421	0.925	−4.212	1.062	−5.308

Table 4
The Luma Y channel BD-BR (%) and BD-PSNR(dB) performance gain of our model on fifty-two Xiph.org [46] video test sequences at different configuration and evaluated with respect to baseline VTM 8.0 framework [38].

Resolution	Sequence		Our model vs. VTM (AI)		Our model vs. VTM (RA)		Our model vs. VVC (LDP)		Our model vs. VVC (LDB)	
	Name	No. of	BD-BR(%)	BD-PSNR	BD-BR (%)	BD-PSNR	BD-BR (%)	BD-PSNR	BD-BR (%)	BD-PSNR
(1280, 720)	Holm	4	−5.402	0.524	−5.379	1.152	−4.642	1.042	−4.883	1.109
	Mobcal	6	−4.017	0.862	−4.195	0.938	−6.456	0.851	−5.541	1.071
	Shields	3	−5.201	0.746	−4.583	0.552	−5.702	1.035	−5.063	1.152
(720, 480)	WashDC	5	−4.257	1.371	−4.604	0.871	−6.053	0.664	−4.856	0.924
	Galleon	4	−3.685	0.852	−4.185	0.897	−4.955	0.902	−5.521	0.842
	Calendar	4	−5.578	0.686	−5.146	0.564	−5.507	1.042	−6.083	1.039
(640, 360)	Fountain	5	−3.716	1.036	−6.372	1.075	−4.748	0.816	−5.474	1.084
	Bridge	5	−5.804	0.851	−4.251	1.08	−5.342	1.037	−4.684	0.762
	Dance	3	−4.806	1.062	−5.392	1.096	−4.687	0.959	−6.086	0.855
(352, 288)	Football	2	−5.107	1.056	−4.238	0.687	−5.244	0.805	−4.932	1.063
	Bus	3	−4.105	0.818	−5.844	1.25	−4.752	0.986	−5.562	1.084
	Bowing	6	−5.839	0.544	−5.352	0.731	−5.304	1.077	−5.511	1.019
	News	2	−4.024	0.694	−6.505	1.035	−4.576	0.925	−5.442	1.167
Average		52	−4.734	0.853	−5.081	0.917	−5.225	0.934	−5.356	1.013

learning framework on CPU–GPU 1080 Ti 11 GB stand-alone system and recorded the decoder time. One can notice from Table 2 that the ARLF [27] based in-loop filtering approach has a time overhead among the three categories in GPU mode. However, benefiting from the GPU acceleration, our D-MSCNN model (12.8%) is 3.9 times faster than the ARLF [27] (50.1%) and got a modest improvement of 2.01 times compared to the MFRNet [28] (38.6%). The above analysis shows that the proposed D-MSCNN network consumes less time than other architectures and configurations and can be a holistic approach in terms of time complexity. However, the decoding time did not meet real-time application requirements and needed further optimization.

4.5. Objective evaluations and results analysis

In lossy video compression, the reconstructed video quality assessment is measured both in objective and subjective performance viewpoints. Our objective video analysis experiment studies the PSNR, VMAF [5], MS-SSIM [32] values for different in-loop deep learning models and compares them at different bit rates. Table 3 summarizes the rate–distortion results among all three approaches in which the original VVC SAO in-loop filtering operation is used as an anchor for baseline. The BD-BR and BD-PSNR results of our algorithm are tabulated in Table 3 for AI, and RA configurations, averaged and

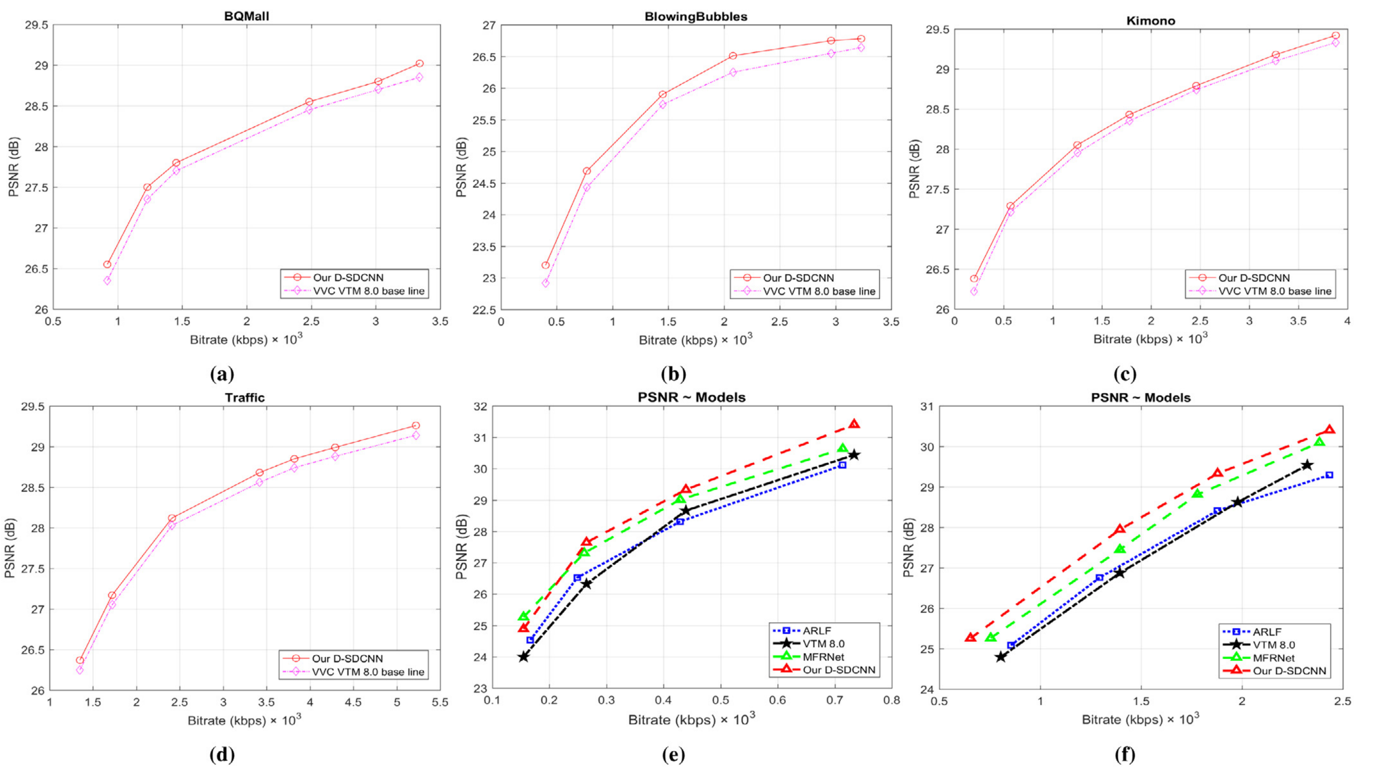


Fig. 5. RD curves for different video sequences and configurations (AI/RA) in ARLF [27], VVC VTM [38], and MFRNet [28]. (a) BQMall, AI; (b) BlowingBubbles, AI; (c) Kimono, RA; (d) Traffic, AI; (e) KristenAndSara, AI mode; (f) BasketballDrill, RA mode.

successively compared with ARLF [27], VTM [38], and MFRNet [28] methods. As reported in Table 3, the BD-BR of our D-MSCNN approach is −5.167% averaged over the conventional test sequences, and outperforming −2.334% of ARLF [27] and −4.142% of MFRNet [28]. We observed that our proposed D-MSCNN performs better both in terms of BD-PSNR improvement and BD bit rate reduction. The MFRNet [28] model outperforms the ARLF [27] because it involves more layers and has higher accuracy. In terms of the PSNR bit-rate match, our approach attains 1.051 dB in the standard test set and considerably well over the 0.707 dB of [15], and 0.883 dB of [28] in AI (intra I-frame), and RA configurations. Eventually, our D-MSCNN model achieved superior rate–distortion performance across all three approaches. The potential reasons for such maximum output include the (1) utilization of wide range activation maps, (2) proposed symmetric convolutional and deconvolutional blocks, and (3) effective feature de-noising and shrinking feature learning. A limited inter-frame experiment was performed with original network settings. Our Conv and De-Conv CNN model is separately applied to different frame blocks in a multi-scale fashion (Fig. 4), reconstructed the image features parallelly [40], and later merged in a gated fusion layer. In our inter coding experiment, each reconstructed frame are served as a reference for encoding successive P or B frames. The CNN model output is then fed back to the next successive frame in a decoded picture buffer to produce its enhanced version. The P and B frames are encoded and trained separately for respective models and followed adaptive QP training strategy addressed in LDP and LDB mode. Additionally, we performed model testing on fifty two Xiph.org video media dataset at [46] and sampled thirteen test sequences from each resolution. Table 4 presents the BD-BR performance of our model under AI, RA, LP, and LB configurations. The proposed architecture shows a stable performance and achieved a −4.734% bit rate saving in AI mode over the video media-set and substantially outperformed other three model configuration results as −5.081% (RA), −5.225% (LDP), and −5.356% (LDB). In terms of BD-PSNR, our LDB configuration model achieved 1.013 dB in the above test set, accomplished 0.853 dB in AI model, 0.917 dB in RA model, and 0.934 dB in LDP mode. In belief, our proposed architecture performs favorably under different compression configurations and network settings and achieves better PSNR values, which demonstrates the effectiveness of our dual hierarchical learning technique.

Fig. 5(a) to (d) show the Rate–Distortion (RD) curves and illustrates the bit-rate savings of our model. The figure additionally shows the PSNR performance comparisons of the proposed D-MSCNN network and H.266/VVC SAO de-blocking filters in luminance frames on four selected sequences: BlowingBubbles, BQ Mall, Traffic sequences, and Kimono. For a fair performance comparison, additional experiments are performed at AI and RA configuration mode in BasketballPass and KristenAndSara sequences (Fig. 5e and f). We used a constant quality of ten bits per pixel to represent each pixel in the frame and measured the number of bits for encoding the representations. The optimal bit-rate in kb/s × 10^3 is calculated by multiplying the above bits per pixel and video resolution. The graphical comparison between PSNR ∼ Bitrate (kbps) is shown in Fig. 5(e) and (f). It can be viewed that the PSNR gain of our D-MSCNN method is higher than that of ARLF [27], VTM [38] methods, and a slight edge in performance to that of MFRNet [28] at different QP values. As shown in Fig. 5(e), the ARLF model output looks inconsistent at higher QP values. The main reason could be, it failed to predict high-frequency wavelet features at different filter sizes and used more memory at the decoder buffer. In brief, the proposed method achieved better feature generalization in our test dataset and obtained consistent evaluation performance under different configurations.

We selected four 4k resolution test sequences of standard Dynamic Range category in UHD resolution [48] and five cropped 8k resolution sequences of different frame lengths from the Fraunhofer HeinrichHertz-Institut [52] and Institute of Image Information and Television Engineers [1]. We provided a rate–distortion comparison in Fig. 6. Five rate points were selected for each test sequence over QP values for the quality assessment. The comparison points have been generated referring to the random access (RA) mode of the VTM-8.0 reference software, calculated average MS-SSIM, and VMA quality metric values. The RD curves for the sequences in Fig. 6 indicate that

Table 5
BD-BR scores of all the models compared to the anchor VTM 8.0 codec. The upper half of the table represents the % bit-rate savings for the quality computed by objective metrics at 8k video. Negative values represent compression gain offered by our model over VVC. The bottom part of the table illustrates the gain in quality regarding for bit-rate for each metrics and at 4k video.

Test sequences		Configuration					
		Bonnineau et. al. [5]		MGNLF [28]		Our model	
		BD-BR MS-SSIM (%)	BD-BR VMAF (%)	BD-BR MS-SSIM (%)	BD-BR VMAF (%)	BD-BR (MS-SSIM) (%)	BD-BR (VMAF) (%)
8k Video Cropped Sequences (7680 × 4320)	OberbaumSpree	−22.32	−25.41	−15.53	−21.06	−26.56	−29.18
	BodeMuseum	−23.05	−27.73	−18.47	−24.01	−27.84	−28.85
	Festival2	−21.36	−26.24	−17.46	−19.41	−24.86	−29.75
	JapaneseMaple	−23.37	−26.86	−15.16	−22.07	−25.88	−30.06
	LayeredKimono	−27.05	−33.30	−21.41	−24.72	−29.88	−35.58
	SteelPlant	−24.40	−26.57	−23.38	−28.44	−28.74	−32.93
Average		−23.59	−27.68	−18.56	−23.28	−27.23	−31.05
4k Video Sequences (3840 × 2160)	NeptuneFountain3	−18.64	−21.52	−16.5	−18.25	−21.76	−25.16
	Marathon2	−17.04	−19.65	−15.04	−17.1	−25.01	−21.85
	MountainBay2	−12.37	−17.34	−9.72	−11.9	−23.41	−22.07
	TallBuildings2	−23.53	−27.89	−20.56	−22.94	−29.07	−31.88
Average		−17.89	−21.6	−15.45	−17.54	−24.81	−25.24

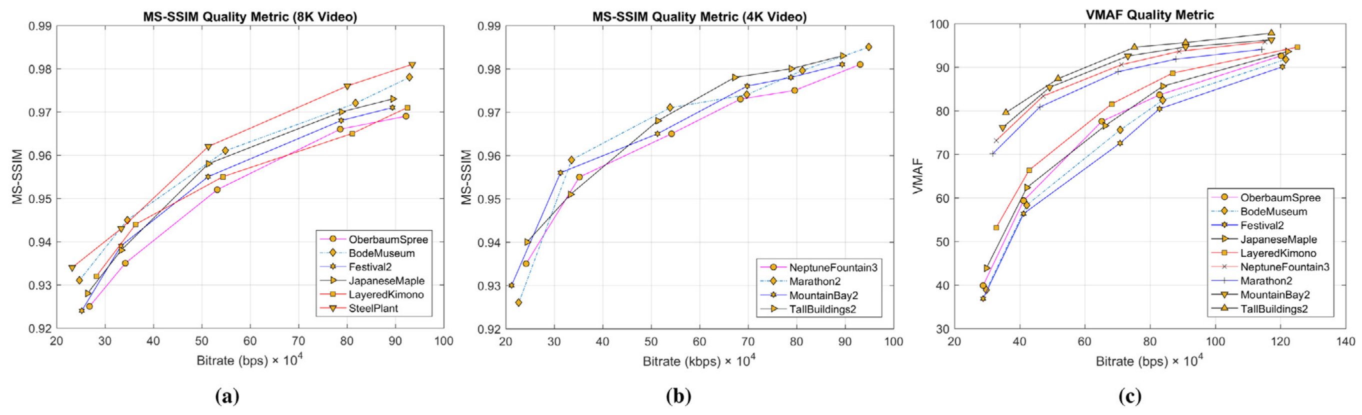


Fig. 6. Objective quality comparison RD plot of our model and its comparison with other models, (a) MS-SSIM for Cropped 8k video sequences@60fps, (b) MS-SSIM for 4k video sequences, (c) Average VMAF and RD curves for different 8k and 4k video sequences.

our model can better deal with irregular object motions and dynamic contents with high-frequency details even at low bit rates. The VMAF is an objective metric that evaluates the perceptual quality between the source and the tested content by scoring between 0 and 100 [5]. This metric is trained to produce a score computed from temporal features across the adjacent frames and achieves a higher correlation across short video sequences and opinion scores (Fig. 6.c). The MS-SSIM compared across multiple frame scales, normalized the luminance and contrast values across the sliding windows, accumulated scopes for each comparison and then averaged to get a frame label quality score [53]. We calculated the average gain in bit-rate offered by other deep-learning models over the VTM-8.0 and presented the average visual quality statistics in Table 5. On average, our model enabled around (24.81%, 25.24%) and (27.23%, 31.05%) bit rate saving over the VVC VTM 8.0 codec, among MS-SSIM and VMAF measures over 4k and 8k video sequences, respectively. The comparison measurements show the VMAF performs better in temporal correlation and is statistically superior to PSNR and MS-SSIM. However, this difference is not statistically significant in higher frame rates and moves these topics for our future studies.

4.6. Subjective results

The subjective video quality assessment requires humans opinions, tedious, but plays an important role in numerous video applications. This section analyzes the decoded frame quality and subjective measurements and subsequently evaluates the perceived visual quality that the human eye may not notice. The subjective study was conducted in a local lab machine setup and used to monitor video quality. The test sequences were visualized on an Ultra High Definition display screen (4k), and students participated in the human perception experiment. Each participant's quality assessment was collected and averaged over each sequence. Fig. 7 exhibits visual image quality comparisons of the original in-loop SAO filter in VVC [38], MFRNet [28], ARLF [27], our D-MSCNN approach, and demonstrate the effectiveness of various architectures. The input "BQMall" sequence has a resolution 832 × 480 and the residual transfer block coefficients are quantized at a factor of QP_{32}. For the quality comparison, we used the fifth frame of "BQMall" video and reconstructed the images. We put a bounding box area around the single person of a zoomed-in region of "BQMall" (right) for small object reconstruction comparison. Although the traditional HEVC filters have been applied in Fig. 7(a), one can notice that there is still ringing and visually blurring effect persists in the baseline results. In Fig. 7(b), the frames filtered by the VVC model look flattened, and an sudden intensity variation is observed through the image objects. Subsequently, one can observe that the reconstructed frames in Fig. 7(c) in the third quadrant (left) are over-smoothed especially in the leg, bag, and shawl region, and introduce noisy lines on the shallow boundary regions. Moreover, some details are missing in the crowded region of the bottom-left corner and minor color distortion at the top right of the elliptical edge regions. On the contrary, Fig. 7(d) refers to the results of the proposed D-MSCNN model. Our proposed adaptive model successfully recovered the feature details around the bag edge, hand and collar regions with less compression artifact. After post-processing, the frame details are restored substantially and look visually appealing, and perhaps this is due to the better generalizability of our model

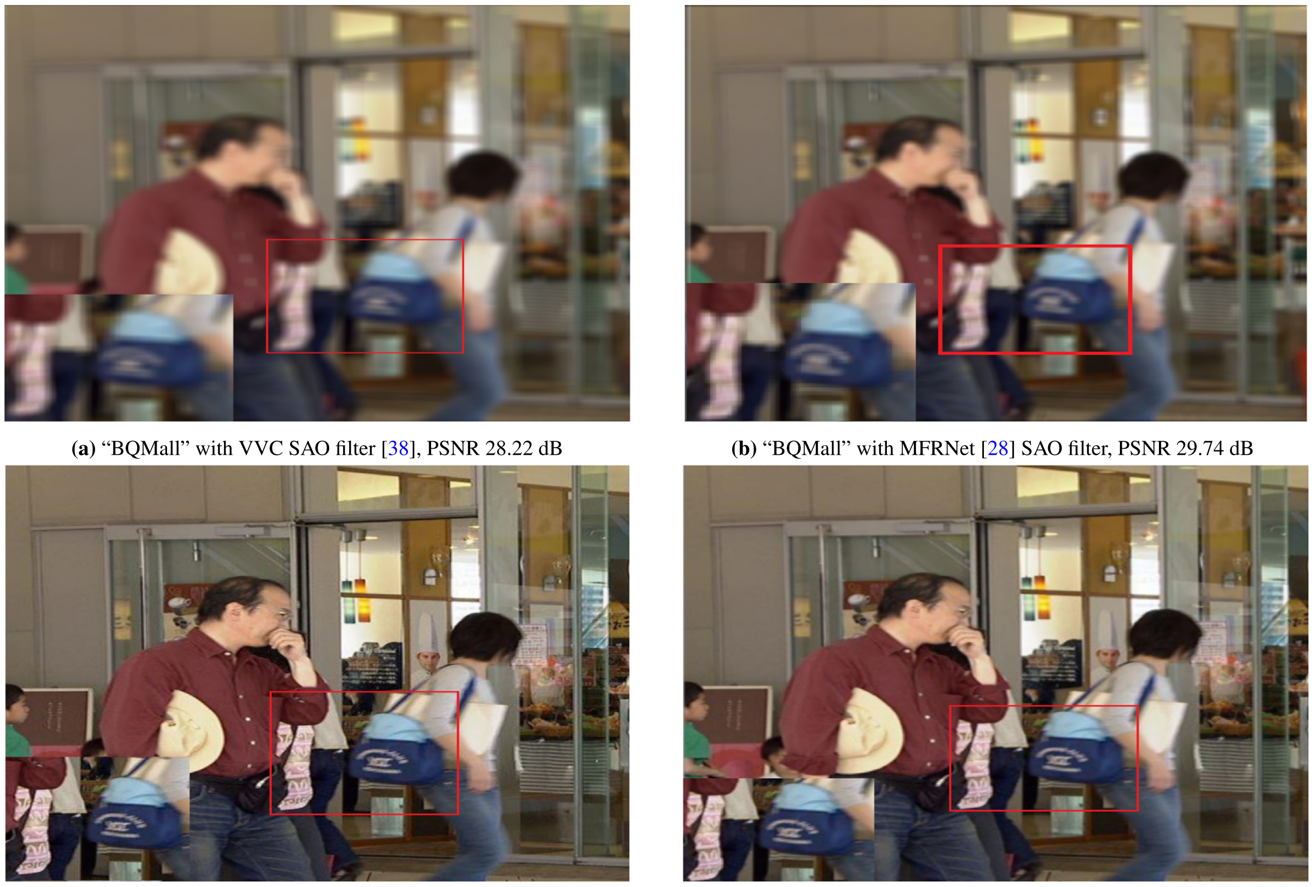

(a) "BQMall" with VVC SAO filter [38], PSNR 28.22 dB
(b) "BQMall" with MFRNet [28] SAO filter, PSNR 29.74 dB
(c) "BQMall" with ARLF [27] SAO filter, PSNR 30.35 dB
(d) Our MSCNN Model, PSNR 31.87 dB

Fig. 7. Visual quality of a compressed "BQMall" with RA configuration and 832 × 480 resolution. The fifth frame with QP = 32 is shown at different SAO filter models.

Table 6
Computation complexity result comparison of different deep learning methods in RA configuration.

Methods	Model size (GB)	GPU memory run time (MB)	Parameters	MACs/pix
ADCNN [29]	38.419	8503	6.535M	498.54
MFRNet [28]	27.065	1942	5.837M	424.03
ARLF [27]	8.370	1503	3.718M	382.61
Our model	6.928	874	0.718M	328.55

Table 7
Ablation study on different model component combinations, and strides (S). BD-BR, BD-PSNR, and VMAF gain on SJTU dataset.

No	Network	BD PSNR (dB)	BD BR (%)	VMAF (%)
1	Conv + Padding + Up-Sampling	1.048	−4.72	−24.26
2	Conv + Un-pooling + De-Conv1	1.427	−5.36	−25.84
3	Conv + Un-Pooling + De-Conv2	1.192	−4.34	−30.09
4	Conv + Un-Pooling + De-Conv3	1.514	−4.93	−28.31
5	Conv + Un-Pooling + De-Conv4	1.375	−5.17	−23.95
6	Conv + Un-Pooling + De-Conv5	0.965	−5.42	−27.73
7	Conv + Average-Pooling + De-Conv5	1.482	−4.18	−31.15
8	Conv + Un-Pool + De-Conv5 + S = 1	1.545	−5.53	−33.91
9	Conv + Un-Pool + De-Conv5 + S = 2	1.472	−5.39	−32.53
10	Conv + Un-Pool + De-Conv5 + S = 3	1.388	−5.26	−29.46

training. However, some details are missing at the shop wall corner and leg of glasses.

4.7. Model computational complexity analysis

The following section reviews the complexity aspects of our model and presents model trainable parameters, multiply-accumulate computation operations per pixel (MACs), run time GPU memory and model memory consumption and compared with other algorithms (Table 6). The average size of the proposed method is 6.92 GB across all network settings. Regarding ADCNN [29], MFRNet [28] and ARLF [27], the model size is 38.41 GB, 27.06 GB, and 8.370 GB respectively. In terms of memory cost, our proposed scheme is superior to MFRNet [28] and ARLF [27] methods and reduced memory costs by 8.73% compared to ADCNN [29] attributed at RA configuration. For the video sequences with the resolution of 832 × 480, the decoding speeds of ADCNN [29], MFRNet [28], and ARLF [27] are 498.54k MACS/pix, 424.03k MACS/pix, and 382.61k MACS/pix, respectively for chroma prediction [14]. It observed that the MACs of our model are comparable with ARLF [27] (382.61k MACS vs. 328.55k MACS) and has a low number of operations per pixel. However, our model is relatively smaller (874M vs. 1503M). The results demonstrate that the proposed network set has lower computational complexity and effectiveness than other recommended networks.

4.8. Ablation study

In this section, we provided a series of ablation experiments with various settings and investigated the performance of our model components. To understand the image enhancement and SAO filter operation, we considered all the output features of all convolution layers for our model testing and altered the deconvolution layers, stride, and pooling (max/average) layers in both upper and lower modules of the dual network. Finally, we came up with ten networks with different components and listed them in the second column of Table 7. All ablation networks used the same training and validation steps for a fair comparison. We report the results on SJTU test Sequences [54] in Table 7 and compared with baseline VTM 8.0. In-network "1", we considered the bilinear interpolation based technique to increase the frame spatial resolution by increasing the number of pixels in both rows and column directions. From Network "2" to "6", we kept all the convolution and up-pooling layers consistent and progressively changed the number of deconvolution layers across the dual network without stride operations. To retain more information about the less important elements of a block, we replaced the max pooling operation with the average pooling In-network "7". Our conventional average pooling layer computed the mean value of the activation maps within a filter window, smooth out the edges and still missed the sharp features. The results show that more coding gain is achieved with network "7" but at the cost of the decoder computation time. Although the average pooling operation achieved robust feature representation in noise, it introduced the blurring effect in object boundaries. One can note that the base Network "8" with all given model components, stride one, and priors achieve the best image enchantment performance in terms of BD-BR, BD-PSNR, and VMAF evaluations. The low PSNR value indicates the blurring and noise amplification effect during the interpolation operation. In networks "9" and "10" we introduced the large stride convolutions and deconvolution operations and estimated the reconstruction quality. It is observed that a larger stride produced a narrower feature map and introduced faster computation time at the cost of a decrease in PSNR performance. One can observe that the activation maps on the fifth Convolution layer (Conv5) embed with rich higher-level features. On the other hand, Conv3 and Conv4 layers have supplementary contextual information, and integrating these feature cues to the deconvolution side benefited the overall filtering operation. The Network "4" and "5" suggest that in the presence of priors, the up-convolution feature learning plays a pivotal role in noise reduction and possibly because of the propagation of fine-grained features during our model learning.

5. Conclusion

This paper proposes a decoupled de-convolution network to enhance the visual quality in reconstructed frames and address the SAO filter post-processing in VVC frame prediction. The dual network utilizes diverse spatial features through skip indexing, fuses the inter-frame feature maps, refines, and recovers the high-frequency details from distorted input. An additional loss component was suggested during the model training time to introduce the image prior features. We first constructed an extensive balanced dataset with different video resolutions for our model evaluation. The experiment results demonstrate that our proposed network outperformed the baseline model and lowered the computation cost. The rate–distortion and the coding efficiency of various state-of-the-art methods are assessed using the same gradient-based optimization techniques. The proposed scheme reduced the compression artifacts at CTU label reconstruction and minimized the rate–distortion cost. Although our network archives promising results in-loop filtering, we plan to analyze the architecture generalization further, remove bias in training data and improve training strategies for real-time usage. Recently, the recurrent neural network has been widely used for video summarization for in-formation retrieval and activity recognition [55]. In the future direction, we would like to explore a vision transformer-based model [56] and non-local attention methods [57] for frame prediction and extend it to LMCS in-loop filter for fractional interpolation learning during chroma prediction.

CRediT authorship contribution statement

Shiba Kuanar: Concept, Model design, Problem analysis, Writing, and manuscript revision. **Vassilis Athitsos:** Concept, Model design, Problem analysis, Writing, and manuscript revision. **Dwarikanath Mahapatra:** Concept, Model design, Problem analysis, Writing, and manuscript revision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The authors do not have permission to share data.

References

[1] ITU-T and ISO/IEC JTC 1, Versatile Video Coding, Rec. ITU-T H.266 and ISO/IEC 23090-3 (VVC), 2020.
[2] B. Bross, J. Chen, J.-R. Ohm, G.J. Sullivan, Y.-K. Wang, Developments in international video coding standardization after AVC, with an overview of versatile video coding (VVC), Proc. IEEE 109 (2021) 1463–1493.
[3] S. Liu, B. Bross, J. Chen, Versatile video coding (draft 10), JVET-s2001, joint video exploration team (JVET), 2020.
[4] G.J. Sullivan, et al., Overview of the high efficiency video coding (HEVC) standard, IEEE Trans. Circuits Syst. Video Technol. 22 (2012) 1649–1668.
[5] C. Bonnineau, W. Hamidouche, J. Fournier, N. Sidaty, J.-F. Travers, O. Déforges, Perceptual quality assessment of HEVC and VVC standards for 8K video, IEEE Trans. Broadcast. 68 (2022) 246–253.
[6] T. Wiegand, G. Sullivan, G. Bjøntegaard, A. Luthra, Overview of the H.264/AVC video coding standard, IEEE Trans. Circuits Syst. Video Technol. 13 (2003) 560–576.
[7] A. Norkin, et al., HEVC deblocking filter, IEEE Trans. Circuits Syst. Video Technol. 22 (2012) 1746–1754.
[8] S.D. Kim, J. Yi, H.M. Kim, J.B. Ra, A deblocking filter with two separate modes in block-based video coding, IEEE Trans. Circuits Syst. Video Technol. 9 (1999) 156–160.
[9] S. Kuanar, K. Rao, C. Conly, N. Gorey, Deep learning based HEVC in-loop filter and noise reduction, Signal Process., Image Commun. 99 (2021) 116409.
[10] Y. Huang, et al., A VVC proposal with quaternary tree plus binary-ternary tree coding block structure and advanced coding techniques, IEEE Trans. Circuits Syst. Video Technol. 30 (2020) 1311–1325.
[11] S. Ma, X. Zhang, J. Zhang, C. Jia, S. Wang, W. Gao, Nonlocal in-loop filter: The way toward next-generation video coding? IEEE MultiMedia 23 (2016) 16–26.
[12] A. Krutz, et al., Adaptive global motion temporal filtering for high efficiency video coding, IEEE Trans. Circuits Syst. Video Technol. 22 (2012) 1802–1812.
[13] J. Kim, et al., Accurate image super-resolution using very deep convolutional networks., in: IEEE CVPR, 2016, pp. 1646–1654.
[14] P. Bordes, F. Galpin, T. Dumas, P. Nikitin, AHG11: Replacing SAO in-loop filter with neural networks, in: JVET-V0092, 2021.
[15] S. Wang, X. Zhang, S. Wang, S. Ma, W. Gao, Adaptive wavelet domain filter for versatile video coding (VVC), in: IEEE Data Compression Conference (DCC), 2019.
[16] M. Karczewicz, N. Hu, J. Taquet, C.-Y. Chen, K. Misra, K. Andersson, et al., VVC in-loop filters, IEEE Trans. Circuits Syst. Video Technol. 31 (2021) 3907–3925.
[17] F. PU, T. Lu, P. Yin, et al., Luma mapping with chroma scaling in versatile video coding, in: IEEE, Data Compression Conference (DCC), 2020.
[18] C. Jia, Content-aware convolutional neural network for in-loop filtering in high efficiency video coding, IEEE Trans. Image Process. 28 (2019) 3343–3356.
[19] C.-M. Fu, et al., Sample adaptive offset in the HEVC standard, IEEE Trans. Circuits Syst. Video Technol. 22 (2012) 1755–1764.
[20] C.-Y. Tsai, et al., Adaptive loop filtering for video coding, IEEE J. Sel. Topics Signal Process 7 (2013) 934–945.
[21] J. Erfurt, W. Lim, H. Schwarz, D. Marpe, T. Wiegand, Extended multiple feature-based classifications for adaptive loop filtering, in: APSIPA Transactions on Signal and Information Processing, 2019.
[22] S. Kuanar, D. Mahapatra, K. Athitsos, V. abd Rao, Gated fusion network for SAO filter and inter frame prediction in versatile video coding, 2019, URL: https://arxiv.org/abs/2105.12229.
[23] T. Li, et al., A deep learning approach for multi-frame in-loop filter of HEVC, IEEE Trans. Image Process. 28 (2019) 5663–5678.
[24] S. Xie, Z. Tu, Holistically nested edge detection, in: IEEE ICCV, 2015, pp. 1395–1403.
[25] D. Liu, Z. Wang, B. Wen, J. Yang, W. Han, T.S. Huang, Robust single image super-resolution via deep networks with sparse prior, IEEE Trans. Image Process. 25 (2016) 3194–3207.
[26] F. Lv, F. Lu, J. Wu, C. Lim, MBLLEN: low-light image/video enhancement using CNNs, in: British Machine Vision Conference, 2018, pp. 1–13.
[27] Z. Huang, J. Sun, X. Guo, M. Shang, Adaptive deep reinforcement learning-based in-loop filter for VVC, IEEE Trans. Image Process. 30 (2021) 5439–5451.
[28] M. Ma, F. Zhang, B. Bull, MFRNet: A new CNN architecture for post-processing and in-loop filtering, IEEE J. Sel. Top. Sign. Proces. 15 (2021) 378–387.

[29] M.-Z. Wang, S. Wan, H. Gong, M.-Y. Ma, Attention-based dual-scale CNN in-loop filter for versatile video coding, IEEE Access 7 (2019) 145214–145226.
[30] VMAF: the journey continues, in netflix technology blog, Oct, 2018, URL: https://netflixtechblog.com/vmaf-the-journey-continues-44b51ee9ed12.
[31] C. Lee, S. Woo, S. Baek, J. Han, J. Chae, J. Rim, Comparison of objective quality models for adaptive bit-streaming services, in: 2017 8th International Conference on Information Intelligence Systems & Applications (IISA), 2017, pp. 1–4.
[32] Z. Wang, E.P. Simoncelli, A.C. Bovik, Multiscale structural similarity for image quality assessment, in: Proc. IEEE Conf. Rec. 37th Asilomar Conf. Signals Syst. Comput, 2003, pp. 1398–1402.
[33] A. Wieckowski, G. Hege, C. Lehmann, B. Bross, D. Marpe, C. Feldmann, M. Smole, VVC in the cloud and browser playback: it works, in: Proceedings of the 1st Mile-High Video Conference, 2022.
[34] Z. Li, A. Aaron, I. Katsavounidis, A. Moorthy, M. Manohara, Toward a practical perceptual video quality metric,'' in Netflix TechBlog, June, 2016, URL: https://netflixtechblog.com/toward-a-practical-perceptual-video-quality-metric-653f208b9652.
[35] R. Rassool, VMAF reproducibility: Validating a perceptual practical video quality metric, in: 2017 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), 2017, pp. 1–2.
[36] N. Barman, S. Schmidt, S. Zadtootaghaj, M.G. Martini, S. Möller, An evaluation of video quality assessment metrics for passive gaming video streaming, in: Proc. 23rd Packet Video Workshop, 2018, pp. 7–12.
[37] C. Zhu, Y. Huang, R. Xie, L. Song, HEVC VMAF-oriented perceptual rate distortion optimization using CNN, in: Proc. IEEE Int. Picture Coding Symposium (PCS), 2021, pp. 1–5.
[38] The VVC test model 8:, 2020, Available:. URL: https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM/-/releases.
[39] V. Badrinarayanan, A. Kendall, R. Cipolla, SegNet: A deep convolutional encoder-decoder architecture for image segmentation, in: IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 39, 2017, pp. 2481–2495.
[40] Y. Cheng, R. Cai, Z. Li, X. Zhao, K. Huang, Locality-sensitive deconvolution networks with gated fusion for RGB-D indoor semantic segmentation, in: IEEE CVPR, 2017, pp. 1475–1483.
[41] F. Bossen, J. Boyce, X. Li, V. Seregin, K. Sühring, JVET-k1010 JVET common test conditions and software reference configurations for SDR video, in: Joint Video Experts Team (JVET), 11th Meeting, 2018.
[42] E. Agustsson, R. Timofte, NTIRE 2017 challenge on single image super-resolution: Dataset and study, in: IEEE CVPRW, 2017, pp. 1122–1131.
[43] CDVL, Consumer digital video library, 2019, [Online]. Available at:, URL: https://cdvl.org/resources/.
[44] VQEG, VQEG video datasets and organizations, 2017, [Online]. Available at:, URL: https://www.its.bldrdoc.gov/vqeg/video-datasets-and-organizations.aspx.
[45] D. Ma, F. Zhang, D. Bull, BVI-DVC: a training database for deep video compression, IEEE Trans. Multimed. (2021).
[46] Xiph, Xiph. Org video test media. [online], 2017, Available:. URL: https://media.xiph.org/video/derf.
[47] F. Bossen, Common test conditions and software reference configurations, in: JCT-VC, Tech. Rep. JCTVC-F900, Torino, Italy, 2011.
[48] M. Wien, V. Baroncini, Report on VVC Compression Performance Verification Testing in the SDR UHD Random Access Category, Tech. Rep. Joint Video Experts Team (JVET) Doc. JVET-T0097, 2020.
[49] T. Nguyen, D. Marpe, Compression efficiency analysis of AV1, VVC, and HEVC for random access applications, in: APSIPA Transactions on Signal and Information Processing 10, 2021.
[50] F. Nasiri, W. Hamidouche, L. Morin, N. Dhollande, G. Cocherel, Model selection CNN-based VVC quality enhancement, in: 35th Picture Coding Symposium (PCS), 2021, 9477473.
[51] F. Bossen, K. Sühring, A. Wieckowski, S. Liu, VVC complexity and software implementation analysis, IEEE Trans. Circuits Syst. Video Technol. 31 (2021) 3765–3778.
[52] B. Bross, H. Kirchhoffer, C. Bartnik, M. Palkow, D. Marpe, AHG4: ''Multiformat berlin test sequences, in: Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 17th Meeting: Brussels, BE, 7–17, 2020.
[53] K. Seshadrinathan, R. Soundararajan, A.C. Bovik, L.K. Cormack, Study of subjective and objective quality assessment of video, IEEE Trans. Image Process. 19 (2010) 1427–1441.
[54] L. Song, X. Tang, W. Zhang, X. Yang, P. Xia, The sjtu 4k video sequence dataset, in: Fifth International Workshop on Quality of Multimedia Experience (QoMEX2013), 2013, URL: https://qualinet.github.io/databases/video/sjtu_4k_video_sequence_dataset/.
[55] G. Bertasius, H. Wang, L. Torresani, Is space-time attention all you need for video understanding? in: Proceedings of the International Conference on Machine Learning (ICML), 2021.
[56] A. Kolesnikov, A. Dosovitskiy, D. Weissenborn, G. Heigold, J. Uszkoreit, L. Beyer, M. Minderer, M. Dehghani, N. Houlsby, S. Gelly, T. Unterthiner, X. Zhai, An image is worth 16x16 words: Transformers for image recognition at scale, in: International Conference on Learning Representations (ICLR), 2021.
[57] R. Yang, et al., Learning for video compression with hierarchical quality and recurrent enhancement, in: IEEE, Conference on Computer Vision and Pattern Recognition, CVPR, 2020.