I.J. Image, Graphics and Signal Processing, **2025, 4, 49-67**
Published Online on August 8, 2025 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijigsp.2025.04.04

Enhancing In-loop Filter of HEVC with Integrated Residual Encoder-Decoder Network and Convolutional Neural Network

Vanishree Moji*
Department of Electronics and Communication Engineering, K S Institute of Technology, Bengaluru, India
E-mail: vanishreemoji@gmail.com
ORCID iD: https://orcid.org/0000-0002-3252-8550
*Corresponding Author

Bharathi Gururaj
Department of Electronics and Communication Engineering, K S Institute of Technology, Bengaluru, India
E-mail: bharathigururaj@ksit.edu.in
ORCID iD: https://orcid.org/0000-0003-2334-5864

Mathivanan Murugavelu
Department of Electronics and Communication Engineering, Salem College of Engineering and Technology, Salem, India
E-mail: mathivananacs@gmail.com
ORCID iD: https://orcid.org/0000-0002-1323-2412

Received: 25 January, 2025; Revised: 12 May, 2025; Accepted: 28 June, 2025; Published: 08 August, 2025

Abstract: High Efficiency Video Coding (HEVC) often known as H.265 is a video compression method that outperforms its predecessor H.264. In HEVC, an in-loop filter is an additional processing step that removes compressing artifacts from decoding video frames while improving visual quality. This research article proposes an improved in-loop filter that incorporates a Residual Encoder-Decoder Network based Deblocking Filter (REDNetDF) and a Convolutional Neural Network based Sample Adaptive Offset (CNN-SAO) filter, which together eliminates the smallest range of artifacts in compression video frames. The quantization frame is subjected to REDNetDF, which removes a minute number of blocking artifacts from the compressed frame. To eliminate the ringing artifacts in the compressed frame, CNN-SAO filter is used. The proposed method is used to evaluate the publicly available UVG dataset. To demonstrate efficiency, the new model is evaluated using a variety of metrics. The outcome of this study provides better results like PSNR of 49.7 dB and the SSIM of 0.97 in comparison with other techniques. Besides, the model's outcome indicates an MSE of 1.8 and saves 24.9% more bits on average to provide the same level of quality as previous techniques. The proposed framework also suppresses time complexities regarding encoding and decoding times with the results of 90.5 and 4.5 seconds on average correspondingly.

Index Terms: In-loop filter, Deblocking filter, Sample Adaptive Offset filter, Residual Encoder-Decoder Network, High Efficiency Video Coding, Convolutional Neural Network

1. Introduction

The Cisco VNI reported that streaming video traffic accounted for 80% of the worldwide web traffic by 2022. Further, with the proliferation of recording devices and the growing demand for panoramic and HD/Ultra HD videos, there is an increasing need for HEVC [1]. HEVC is a video coding standard developed in collaboration with VCEG and MPEG [2]. HEVC retains the core hybrid coding design of prior video coding standard like H.264/AVC. Unlike H.264/AVC, HEVC uses a coding tree unit (CTU) instead of a macroblock [3, 4]. In block-based video coding, each block is compressed and transformed independently, which can cause discontinuities around block boundaries in the reconstructed image [5]. Encoder design balances various criteria such as simplicity, coding efficiency, data recovery, and minimal delay, while decoders must produce defined outputs from coded bitstream without being constrained to the same operations [6].

HEVC applies a selective deblocking filter to reduce minor artifacts, and its in-loop filtering mechanism removes blocking artifacts and chromatic aberrations, thereby improving prediction accuracy and overall image quality [7, 8]. HEVC's Context-Adaptive Binary Arithmetic Coding (CABAC), adapted from H.264/AVC, offers excellent coding efficiency but presents challenges for parallelization and throughput [9]. The coding randomness in HEVC strikes a balance between compression effectiveness and processing speed, enhancing performance and data reduction. Highly compressed video often exhibits artifacts such as block boundary degradations, corner outliers, and ringing noise [10]. These artifacts can spread due to motion estimation, complicating detection and increasing deblocking costs compared to in-loop filtering, which is applied only at block boundaries [11].

The major contribution of this research article:

- Improving Deblocking filter in the HEVC in-loop filter with the Residual Encoder-Decoder Network that minimizes coarse and fine grained video artifacts.
- The Sample Adaptive Offset filter eliminates ringing distortions from compressed video frames during coding. The SAO filter has been combined with the Convolutional Neural Network to enhance efficiency and remove the varied artifacts present in the compression frame.

The sections of this research article are organized as: Section II offers a summary of existing studies, while Section III depicts the methodologies for HEVC filtering modules. The proposed approaches are discussed in Section IV. Sections V and VI presents the experimental results and conclusions respectively.

2. Related Studies

Unfortunately, when the pattern is utilized to enter the video coding system, the rebuilt frames contain chromatic flaws and other irregularities. The following section addresses existing research on artifact removal along with methods for HEVC.

Xin Lu et al. [12] suggested a hybrid solution for HEVC intra coding that reduces computing complexity by combining rapid CU size determinations with quick PU mode decision making procedures. It uses algorithms that adapt to evaluate video material homogeneity, narrow depth ranges, and evaluate mode correlation. The main limitations accompanied with this study was low generalizability to the wide range of video content, insufficient investigation of the trade-off between increase in complexity and improvement in video quality, poorly defined guidelines for adaptive thresholds, insufficient examination of the interrelation between coding modes parameters, and limited comparison with the existing fast coding.

To overcome these challenges of the in-loop filter in HEVC and artifact removal, Zhaoqing Pan et al. [13] developed EDCNN using a weighted normalization technique, an attribute data combination block, and an accurate loss function. The developed EDCNN based in-loop filtering algorithm proposed in this paper exhibits a potential in improving the PSNR performance, but in the real time application, difficulties may arise from computational time and GPU memory usage. However, the study targets the organization of the EDCNN model primarily and maybe useful in understanding its effectiveness in solving some of its real life problems.

Qiang Xu et al. [14] created an IFPP method that detects video transcoding by utilizing in-loop filtering and PU partition features. This work focuses on recognizing video transcoding using a novel technique that incorporates a 17 dimensional classifying feature. The paper fails to analyze the practical real time implementation of the developed algorithm which is important for real world application like video tampering and video content authentication. This may be a limitation, meaning that more outcomes from the algorithm needs to be studied on using them for real world applicability of real time transcoded HEVC videos.

Dhanalakshmi A. and Nagarajan [15] G. developed a CResNet architecture with an in-loop filter to decrease visual artifacts. It contains four convolution layers for the basic and enhancement layers in GOP film. Data for training is prepared using the compression sequence with various Quantization Parameters (QPs). The architectural design seeks to increase video compression speed by using separate layers for different aspects of filtering, leading to a better overall appearance and efficient artifact elimination. There is also a consideration that the efficiency of the proposed method depends on the type of the video content. Some sorts of content can be less enriched by the spatial temporal filtering, which can negatively reflect on the result for different video sorts.

Prayline Rajabai Christopher et al. [16] developed the framework to test dual parallel edges DF architecture for HEVC that preserves visual quality when processing video applications. The study investigates the impact of optimization strategies on the utilization of equipment, such as lowering electricity consumption and algorithmic complexity, while ensuring video quality is not compromised. The developed study did not delve into the specific hardware implementation challenges or provide a detailed comparison with existing deblocking filter architectures in terms of computational efficiency or cost effectiveness.

Su Yeon Lim et al. [17] proposed a strategy to improve fraction interpolation for intra prediction in VVC. It introduces new interpolated filters that employ more integer positioned sample references based on their frequency properties. It generates two 8-a-tap interpolated filters with improved high and low frequency filtering properties, yielding more accurate fractional forecast samples. Thus, the dependence of the proposed method on high frequency

ratio from the 1D scaled DCT may not be as effective under the condition of varying frequency characteristics. In terms of the real time application, a higher computational degree (2% for encoder and 5% for decoder) was observed. Less testing across JVET conditions coupled with sub optimal selection of the threshold even more limits generalization and robustness across different scenarios.

Thanuja Mallikarachchi et al. [18] present a CTU level decoder with complexity and rate control for HEVC. To lower the total cost of rate, decode complexity, and distortions for a specific QP. Peculiarities of the study include the use of highly tuned parameters for CTU level coding, which might complicate the adaptation of a solution for various cases. Furthermore, while lowering energy utilisation, the algorithm's integration may prove difficult particularly for devices with limited hardware or low power.

Most in-loop filtering techniques applied in HEVC face challenges of achieving a right balance between computational complexity, effectiveness in removing artifacts, and flexibility for a wide of range video contents and use cases [19]. According to the above analysis, the traditional approaches have drawbacks such as low generality, fewer attempts at balancing between intricacy and video quality, and insufficient tuning of the parameters, and applicability in real time settings. These limitations limit their usability for practical artifact removal cases, such as blocking and ringing distortions but they are computationally efficient. However, current in-loop filtering methods suffer from some of the above limitations, and hence, an efficient in-loop filtering methodology is required to address these problems and improve quality measures of videos and at the same time meet scalability, stability and low computational complexity challenges. To address previous research limitations, we replaced the DF with a REDNetDF and enhanced the SAO filter with a Convolutional Neural Network, resulting in a significant reduction in ringing artifacts, as well as facilitating CTU based decoding and encoding procedures and lowering hardware implementation costs.

3. Methodology

In this study, we propose a Deblocking filter based on REDNet combined with a CNN-based Sample Adaptive Offset (SAO) filter to address spatial and temporal distortions in HEVC [20]. Input video frames to be encoded are divided into two blocks: Coding Tree Units (CTUs) and Coding Units (CUs). Each frame is segmented into CTUs based on its grid, which can be further partitioned into CUs up to 64x64 pixels. Further, CUs are classified into Transform Units (TUs) and Prediction Units (PUs) [21]. HEVC employs two main types of predictions-intra frame and inter frame. Intraframe prediction exploits spatial redundancy by using adjacent pixels within the same frame, applying multiple directional modes. Interframe prediction leverages temporal redundancy by estimating motion vectors to predict the current block based on previous frames [22]. The residual difference between actual and predicted blocks is transformed via Discrete Cosine Transform (DCT) or Discrete Sine Transform (DST), then quantized to reduce precision and compress data [23]. Finally, quantized coefficients and motion vectors are entropy-coded using Context-Adaptive Binary Arithmetic Coding (CABAC) or Context-Adaptive Variable Length Coding (CAVLC). The in-loop filter implies that filtering occurs as a component of a blocking artifact removal process [24].

Hence, the proposed in-loop filtering framework, illustrated in Fig. 1, integrates REDNetDF and CNN-based SAO filter to enhance artifact removal in HEVC. REDNetDF operates via encoder-decoder architecture, extracting features from compressed frames and reconstructing artifact-reduced frames. The CNN-based SAO filter complements this by predicting adaptive pixel-level offsets to reduce ringing artifacts, effectively enhancing coding tree units. Together, these modules comprehensively address artifact removal, improving video quality without excessive computational cost. Fig. 2 presents the combined architecture, showing how REDNetDF's residual blocks, convolutional noise reduction, dense feature connections, and residual attention mechanisms work in tandem with the CNN-based SAO filter. The SAO filter predicts residual corrections via convolutional layers followed by a sigmoid activation, and the corrections are added to the decoded picture buffer for final enhancement. This integrated approach offers a robust solution to improve the visual quality and realism of HEVC compressed videos.

3.1. REDNet based Deblocking filter

In this paper, a REDNet based Deblocking Filter is illustrated in the Fig. 3, which aims to eliminate the blocking artifacts in the compressed video frames. To effectively detect artifacts and capture multi scale spatial features, yet light weighted, the encoder employs convolutional layers and attention blocks as mentioned in Table 1. The decoder recovers the frame using these features by employing upsampling and residual connections and improves the details with a Channels Weighted Block (CWB). Further, the Residual Refinement Network brings higher accuracy because learned artifacts areas use deep supervision for minimizing distortions. This architecture combines hierarchical feature extraction with accurate reconstruction and it shows improvement in blocking artifacts and video quality.

Table 1. REDNet based Deblocking filter architecture parameters

Component	Parameters / Description
Encoder – Convolutional layer 1	64 filters, kernel size 3×3, stride 1, padding 1, ReLU activation
Encoder - Convolutional layer 2	64 filters, kernel size 3×3, stride 1, padding 1, ReLU activation
CBAM Attention	Channel and spatial attention with shared MLP (128, ReLU, 64)
Context Module	Dilated conv layers with dilation rates = {1, 2, 4}, kernel = 3×3
Residual Connections	Skip connections at each encoder-decoder level to preserve gradients

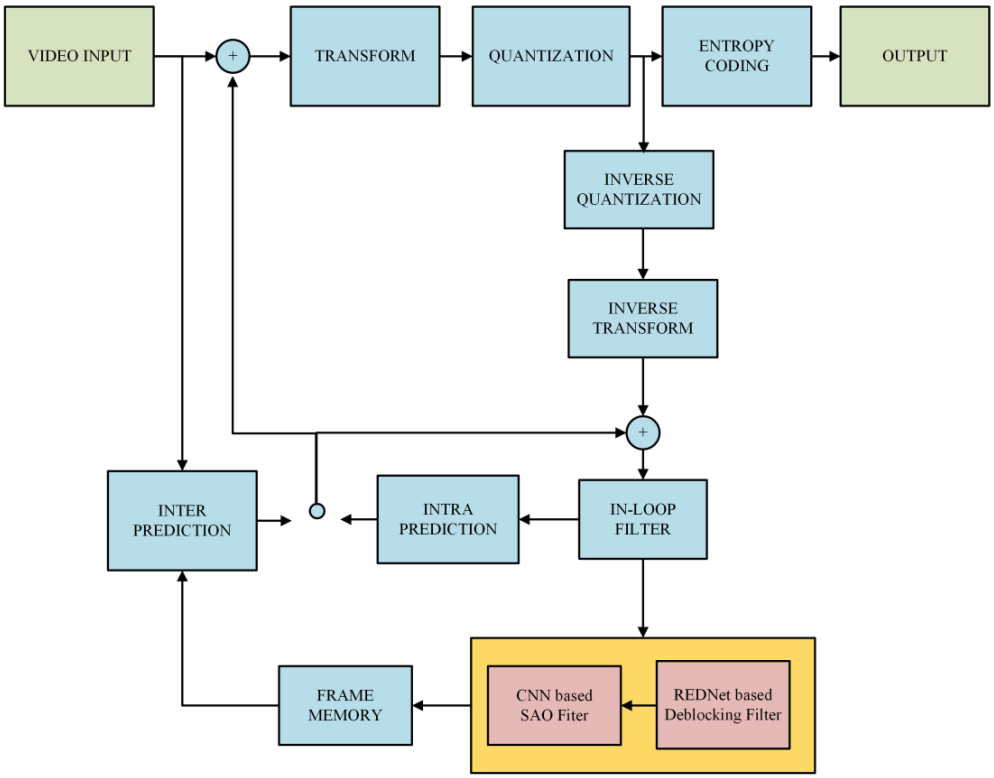


Fig. 1. Design frame work of proposed model

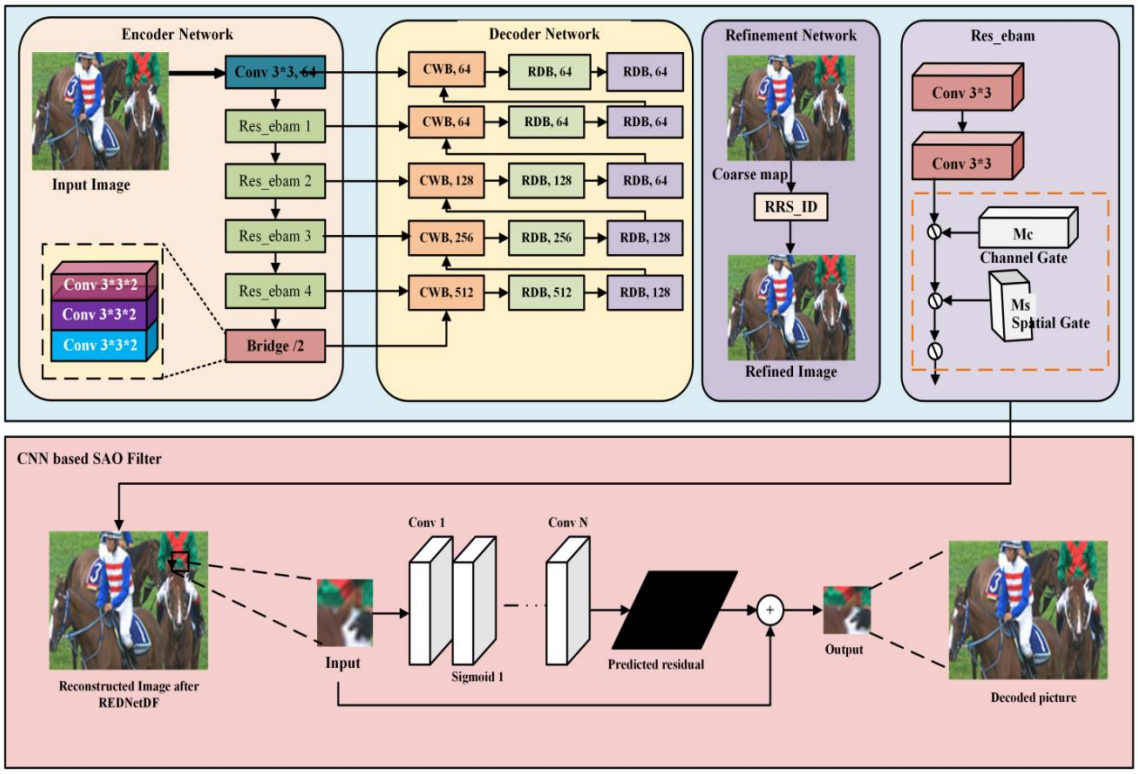


Fig. 2. Conceptual frame work of proposed model

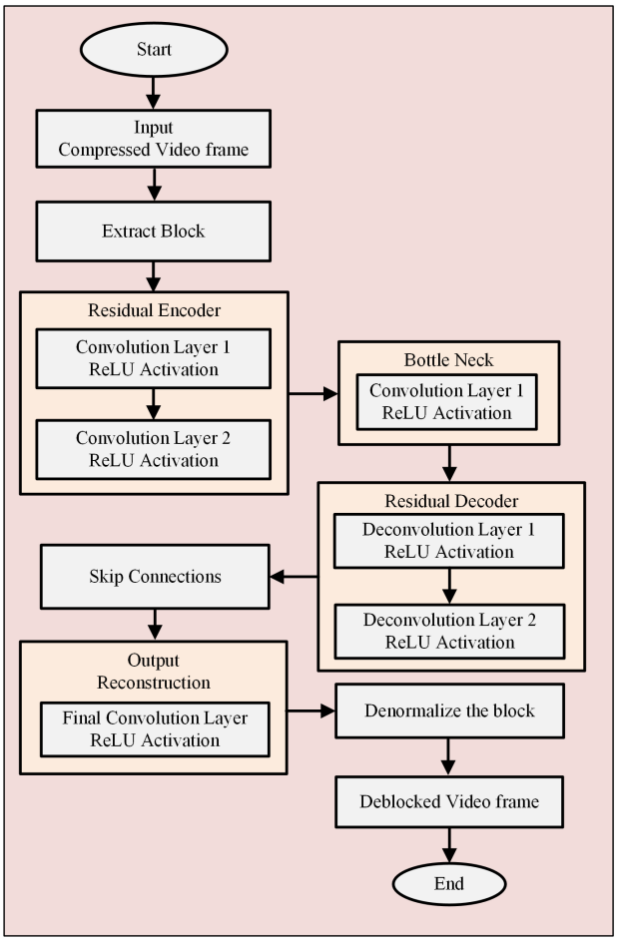


Fig. 3. Flow diagram of proposed REDNet based Deblocking filter

3.1.1. Encoder network

The network setup effectively integrates broad spatial data with the perception of context, raising the precision of artifact identification. The encoder component includes an initial layer of combination with 64 channels and a 3×3 kernel size, with a stride of 1, instead of 7×7 with stride 2. The dimension of the reception field is then increased in two steps using a max pooling technique. Each convolutional conclusion is batch adjusted to guarantee that the characteristic sizes are constant before being passed to a code that triggers Rectified Linear Unit (ReLU) to boost complex encoding capabilities. Given an input image, $I \epsilon \mathbb{R}^{HXWXC}$ represents its height, width, and number of channels, correspondingly. We can extract characteristics using multiple scales of six levels, denoted by *{f, i = 0, 1, 2, 3, 4, 5}* and resolutions of $[H/2^i, W/2^i]$. The method's other basic elements each include a lightweight convolution block attention module (CBAM). As a result, we created a further connection module that collects global context aware data. This module uses three 512 transmit 3×3 convolutional layers with $\{r = 2^m, m=0, 1, 2\}$ dilations, where r represents the dilation rate. To keep the feature map's resolution, the first layer of convolution is downsampled with a stride of two. The sequential normalization and ReLU activation algorithms are then applied to each convolutional layer in sequence.

3.1.2. Decoder network

In the decoding system, we used the CWB and Residual Decoder Block (RDB) to gradually reclaim saliency data from previous multi scale features. It works by merging feature maps: currently encoding feature X and the resulting feature Y from the next decode step, which is upsampled by a factor of two to correspond to X's quality.

GAP generates channel wise vectors ($F_c \epsilon \mathbb{R}^{CX1X1}$), enhancing global knowledge and consistency. A bottleneck structure with two 1×1 convolution layers and a Parametric ReLU activation function reduces the complexity of the model and increases adaptability. A function with a sigmoid coefficient is then applied to the previous output, yielding a weight matrix P in the range [0, 1]. Finally, utilizing the advantages of residual the final CWB result is computed by combining the improved value X with the initial deeper characteristics Y using (1),

$$\begin{cases} P = \sigma(fu_{conv}(GAP(Conc_{up}\,(Y, X)))) \\ Z = Y \oplus (X \odot P) \end{cases} \quad (1)$$

where GAP denotes global average pooling, fu_{conv} is the bottleneck structure of feature fusing, σ is the sigmoid activation function, $\oplus$ represents an element wise summing operation, $Conc_{up}$ means upsampling concatenating operation, $\odot$ denotes an element wise multiplication.

As mentioned in Table 2, the decoding system employ channels flip between two 3x3 convolutional layers. A process like this can boost generalization capabilities and find more theoretically beneficial data. Two 3x3 convolutional layers were used, followed by normalizing and stimulating with Parametric ReLU functions. We add a 1×1 convolutional layer for two different goals.

Table 2. Decoder - Context Weighted Block (CWB) and Residual Decoder Block (RDB)

Block	Parameters / Description
CWB	1×1 Convolutional layer s+ Global Average Pooling + ReLU + Softmax scaling
RDB - Convolutional layer 1	64 filters, kernel 3×3, BatchNorm + PReLU
RDB - Convolutional layer 2	64 filters, kernel 3×3, BatchNorm + PReLU
Channel Flip	Alternates channel-wise features to diversify learning
Final Layer	1×1 Convolutional layer, no activation (linear output)

3.1.3. Residual refinement network

Deep monitoring is utilized during training to confirm feature conformance to actual data at the final convolutional layer in each step of the REDNet. This allows for the efficient propagation of relevant info to problematic regions in input images. The maps are generated using a single channel 3x3 convolutional layer, followed by bilinear expanding and sigmoid stimulation for output in the range [0, 1]. Notably, the ultimate saliency map has higher detection accuracy and more prominence details, making it an excellent input for the refinement networks.

The bridge segment utilizes a 64 channel 3×3 convolutional layer, sequential normalization, and ReLU activation. To reduce computational complexity, we employ specialized 1D filters (3×1 and 1×3) rather than 3×3 convolutions. Convolutional dilation is also employed to extend the receiving field, improving accuracy while increasing processing complexity. These solutions combine to lower data transfer and computing demands in our framework.

3.2. CNN based SAO filter

The initial HEVC encoder includes an additional filter called SAO. The filter uses the same offset for each CTU, whereas the CNN based SAO lowers distortion by automatically computing different offsets for each pixel within a single CTU. Our suggested strategy reduces distortions more effectively than typical SAO. On the other hand, we use the CNN based SAO filter model illustrated in Fig. 4 with equal weights for both the decoding and encoder components, therefore CNNSAO filter does not require an input signal from the decoder. As listed in Table 3 and Table 4 we selected a simple CNN with five layers based SAO network design after analyzing variety of contrast events. The first to fourth convolutional layers have 32 3x3 filters each; however, the final layer only contains one. The first through fourth convolutional layers are activated using the sigmoid activation technique.

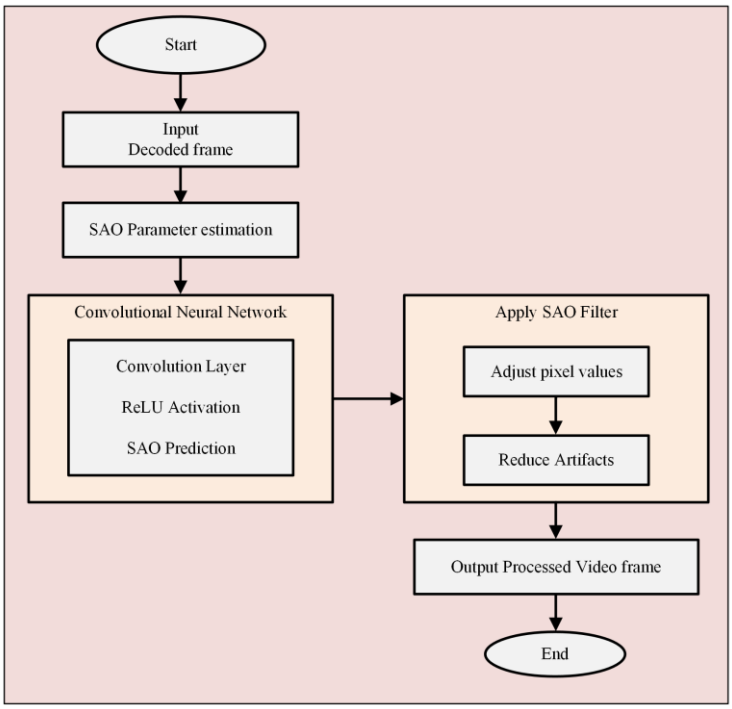


Fig. 4. Flow diagram of proposed CNN based SAO filter

Table 3. CNN based SAO filter design parameters

Layers	Parameters
Convolutional layers 1 to 4	32 filters, 3×3, stride 1, padding 1, Sigmoid activation
Convolution layer 5 (Output layer)	1 filter, 3×3, stride 1, padding 1, Linear (no activation)

Table 4. Thresholds and filtering decisions

Decision Point	Strategy / Thresholds used
Blocking artifact detection	Feature deviation > adaptive threshold (learned via CNN)
Pixel-wise Correction	Offset generated per-pixel by CNN output, scaled via Sigmoid
Offset Application	Conditional replacement if offset > 0.01 (empirically selected)
Loss Function	MSE Loss + L1 penalty on correction residuals

4. Proposed REDNetDF and CNN-SAO filter

The conceptual model of the proposed framework is illustrated in Fig.1 to Fig.4. The Proposed REDNetDF integrates a Residual Encoder-Decoder Network with a CNN based SAO filter to enhance video quality by removing artifacts which are discussed below:

4.1. Encoder layers

To recover artifact related information from compressed video patches, turn them into latent representations. The upper half of the network contains the encoder, which is made up of two convolution branches. The technique begins with two downsampling operations: a 7×7 convolution layer with stride 2, and a 3×3 maximum pooling layer with stride 2. This max pooling layer is the only architectural pooled layer; all other downsampling and upsampling procedures are accomplished with two stride and transposition convolutions.

Following the first levels, the encoder contains residual layers with variable numbers of remaining units. Notably, only the first layer of the encoder lacks a downsampling unit; all succeeding residual layers have a single residual unit that downsamples and repeat the distinctive map channels. The feature fusion method uses element wise summarizing using (2), (3) and (4) mentioned below:

$$A(v) = \sum_{i=1}^{5} \emptyset(v_i - v_{i+1}) \tag{2}$$

where,

$$\emptyset(\Delta) = 0, for\ |\Delta| \leq S \tag{3}$$

$$\emptyset(\Delta) = 1, for\ otherwise \tag{4}$$

Here, to decide the mode of filtering, smoothness or complexity of a region in an image $A(v)$ is compared with two thresholds $T_1\ and\ T_2$. The value of the threshold T_1 is set to a small value. To decide an essentially smooth region the value of the threshold is set to the value to decide a complex region. If $A(v) < T_1$ and $abs\ (offset) < QP$, then smooth mode filtering; else if $A(v) < T_2$ and $abs\ (offset) < QP$, then complex mode filtering and else for $T_1 < A(v) < T_2\ and\ else$, then intermediate mode filtering. $A(v)$ denotes the aggregated measure of smoothness or complexity in a region, computed from differences between neighboring pixel value; v_i denotes the value (e.g., intensity or feature) of the ith pixel or element in the region; v_{i+1} denotes the value of the next pixel or element adjacent to v_i ; $\emptyset(\Delta)$ denotes the thresholding function applied to the difference $\Delta = v_i - v_{i+1}$; T_1 denotes the lower threshold for classifying a region as "essentially smooth"; T_2 denotes the higher threshold to classify a region as "complex"; $abs\ (offset)$ denotes the absolute value of the filtering offset or adjustment factor applied to the pixel or region; and QP denotes the quantization parameter representing allowable error or distortion limit.

4.2. Residual blocks

Residual connections (residual blocks) are key components of remaining systems, letting the network focus on learning the remaining connections. They increase learning by highlighting residual mappings (differences) across the whole input to output mapping. Each residual block typically starts with two convolutional neural networking layers, followed by batch normalization, ReLU activation, and Identity Mapping. The residual blocks that conduct the mechanism of the deblocking filter are provided below in (5), (6), (7) and (8):

$$S_1 = \sum_{\substack{i=1 \\ i \neq 5}}^{\bar{9}} \propto_i P_i \tag{5}$$

where

$$\propto_i = \begin{cases} 1, & |P_5 - P_i| < TH \\ 0, & otherwise \end{cases} \quad (6)$$

$$S_2 = \sum_{\substack{i=1 \\ i \neq 5}}^{9} \propto_i P_i \quad (7)$$

$$P_5' = (\beta P_5 + S_1) / (\beta + S_2) \quad (8)$$

where *TH* is a set based on the parameter of quantization. In smoothing, *TH= QP* is set if P_5 is within the inter coded block, and *TH= QP/2* if P_i is within the interceded block, with β falling within the minimum and maximum range of controlling extent, $\propto_i$ Weight factor of the pixel value at position i , P_5' represents the filtered value of the center pixel and S_1 and S_2.

4.3. Decoder layers

The second half of the network, beginning with the Trans1 layer, functions as a decoder. Apart from the last convolutional layer, which is a single 2 ×2 transpose convolutional layer, the decoder's layers are all residual. The initial four layers (Trans1, Trans2, Trans3, and Trans4) each have one upsampling residual unit, which multiplies the feature map by two. Despite the encoder's bottleneck building blocks, the decoding algorithm uses normal residual components and two consecutive 3 ×3 convolution layers for residual calculation.

In terms of upsampling processes, the decoder employs upsampling residual units. The residual layer output in the residue encoder has a large channel size as a result of channel expansion. Agent layers (single 1 ×1 layers of convolution with stride one) are used to move the feature map to a smaller channel size and reduce the amount of memory utilized in the decoder. Equations (9) to (11) are used for the process and are provided below:

$$offset = P_3 - P_4 \quad (9)$$

$$P_i' = P_i - sign\ (offset) * \left(\frac{|offset|}{\propto_i}\right)' \ for\ i = 2,3 \quad (10)$$

$$P_i' = P_i + sign\ (offset) * \left(\frac{|offset|}{\propto_i}\right)' \ for\ i = 4,5 \quad (11)$$

where P_i represents the interceded i^{th} block, and P_i' represents the interceded i^{th} block.

The order of residual units varies between upsampling and downsampling ResLayers. The downsampling layer starts with a downsampling residual unit and goes through several residual units, while the upsampling layer starts with several residual units and ends with an upsampling residual unit.

4.4. CNN layers of SAO

The DF's final output serves as an input for the CNN based SAO filter. It takes the decoded frame as input and uses convolutional layers to carefully examine local patches within the frame.

4.4.1. Edge Offset

The Edge offset (EO) unit uses statistical approaches to calculate deviations for each EO class group. It consists of two steps: CNN processing and edge category generation. The current sample is at the L1 (1) position. CNN layers identify EO classes 0 (horizontal) as white circles, 1 (vertical) as black circles, 2 (135 °) as dotted circles, and 3 (45 °) as circles with dashed lines. This design generates offsets for deblocked samples by doing statistical computations on each of the four EO groups and their associated categories. This enables offset computations after the CTU process. A single category of each EO class is altered at a time, and if the category is 0, no further modifications are made.

4.4.2. Band offset

The Band offset unit uses CNNs to divide data into 32 bands and execute statistical computations to provide exact offsets. CNN layers improve classification accuracy by examining spatial and contextual aspects of the data. Following classification, the unit collects data to calculate offsets, which account for variances in the deblocked samples. This CNN processing integration enables accurate band classification and efficient offset generation, which improves the adaptive offset filter's overall performance.

These convolutional layers look at the distribution and specific aspects of the artifacts present. Based on this analysis, the CNNSAO filter calculates offsets or modifications to pixel values to effectively decrease distortions. The CNNSAO filter generates a filtered version of the input frame, with artifacts greatly reduced or removed. This technique significantly improves the visual clarity of the video frame, bringing it closer to the original unprocessed

information. Integrating CNNSAO filter into the in-loop filter of H.265/HEVC video codecs ensures that these enhancements occur smoothly and efficiently while adhering to industry requirements for enhanced video compression and decompression methods. This is indicated in (12).

$$Corr(s) = \sum_{i=0}^{M-1} w(s)_i . off_i \tag{12}$$

where, off_i represents the transmitted offset parameters, $w(s)$ denotes the weight factor for every block of compressed video frame, *S* denotes the block of the compressed frame.

5. Results

The REDNetDF and CNNSAO filter for the HEVC standard are evaluated. Various test video sequences are utilized to calculate the PSNR values for the REDNetDF and CNNSAO filter design at different QP values. The original/raw video sequence from the UVG dataset is fed into HEVC.

5.1. Dataset description

There are sixteen test video sequences in the suggested UVG dataset. They were recorded with a Sony F65 video camera at 50 or 120 frames per second in 16-bit F65RAWHFR format, and FFmpeg was used to convert them to 10- and 8-bit 4:2:0 YUV videos [25]. The ITU's suggested spatial perceptual information (SI) and temporal perceptual information (TI) ratings are then used to characterize our dataset. Further, the most recent version of HEVC/H.265 reference encoder use default 10bit configuration of HM16.20 is used to assess the RD features of these sequences. Furthermore, the suggested complexity metric is used to evaluate the sequences. Table 5 provided below displays the video frame input to HEVC, as well as the artifacts that decreased the UVG video sequences and the frame rate.

5.2. Experimental setup

This section examines the feasibility of a formulated algorithm for regulating the decoding complexity and rate at the CTU level. It then compares the rate and complexity controlling capabilities of the proposed algorithm to that of two recently proposed decoding complexity aware encoding algorithms from the literature. Following this, it provides the experimental outcome of the decoder's power consumption during the streaming of video under two different methods of frequency control over the CPU. In addition, the final part of the section consists of the authors' reflections on the experiments and observations made within different applications.

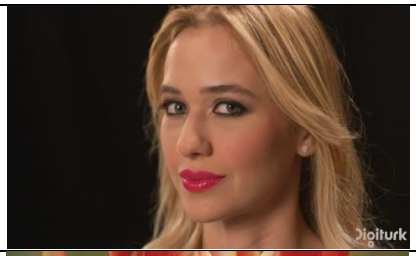
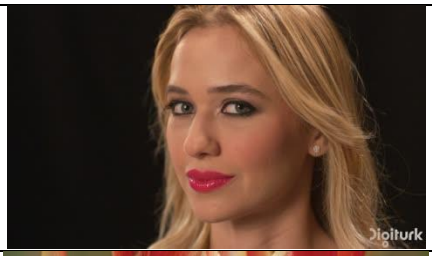
5.3. Performance metrics

Herein video sequences are segmented into smaller processing units (blocks or frames) and each unit is evaluated for the presence or absence of compression artifacts specifically blocking and ringing. The detection results are compared with the ground truth (original video) to determine classification outcomes: True Positives (correctly detected artifacts), False Positives (falsely detected artifacts), True Negatives (correctly identified non-artifact regions), and False Negatives (missed artifacts).

From these outcomes:

- Accuracy reflects the overall capability of the model to correctly identify both artifact and non-artifact regions across frames.
- Precision evaluates how many of the detected artifacts were actually correct, signifying the model's reliability in avoiding false alarms.
- Recall (or Sensitivity) indicates how effectively the model captures all existing artifacts, showing detection completeness.
- Specificity helps assess how well the model avoids false positives in clean areas, particularly relevant to ringing artifacts.
- For Bitrate Reduction, we calculate the percentage change between original and compressed bitrates, which quantifies the model's efficiency in reducing file size while maintaining quality.
- Computational Time is recorded for encoding and decoding processes to assess the model's feasibility for real time or near real time applications. A comparison of expected and actual processing times highlights computational efficiency.

Table 5. Illustrates the coding sequences

Encoding Video Sequence	Frame Rate	Artifacts Reduced Video Sequence
	120fps	
	50fps	
	50fps	
	120fps	
	120fps	

Accuracy: Accuracy is employed to assess the efficiency of the proposed model in detecting and eradicating blocking artifacts in the samples. It is computed using (13):

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \tag{13}$$

where True Positive (TP) mean correct detection of blocking artifacts mean. False Positive (FP) signifies wrongly pointed blocking artifacts. True Negative (TN) is defined as the number of false positive which correspond to non-blocking artifact but are correctly classified. False Negative (FN) is defined as missing blocking artifacts.

Precision: Precision assesses the effectiveness of the proposed strategy for detecting artifacts. It is determined using (14):

$$Precision = \frac{TP}{TP+FP} \tag{14}$$

Recall/Sensitivity: Recall or Sensitivity is used to detect the existence of blocking artifacts in video sequences and is calculated using (15):

$$Recall = Sensitivity = \frac{TP}{TP+FN} \tag{15}$$

Specificity: Specificity measures the occurrence of ringing artifacts in video sequences. It is calculated using (16):

$$Specificity = \frac{TN}{TN+FP} \tag{16}$$

Table 6. and Fig. 5 illustrate the performance of a model across five sources (Beauty, Flower Focus, Flower Pan, Honey Bee, and Jockey) using key metrics: Sensitivity, Specificity, Positive predictive Value and Negative Predictive Value. They usually maintain a high degree of accuracy with values that range from 94–96% this is therefore a good representation of the overall prediction. Precision is slightly lower (between 92% and 98%), with Flower Focus scoring the best mark but slightly worse recall of 93%, lending fewer false positives but lost true positives. Positive Predictive Value, (93–97%), demonstrates the model's capacity to detect most of the positive existence, whereas Negative Predictive Value (94–97%) helps to discover true negatives efficiently. Among these sources, Flower Pan has the highest values and the best balance of the prominent metrics. With reference to Precision, Jockey and Honey Bee exhibit a slightly lower score, indicating that several results are not accurate and are, therefore, false positives. Altogether, the model is robust and followed the same pattern without much post calibration required except for Flower Focus and Jockey where Precision and Recall can be further uplifted a little.

Table 6. Performance of the proposed model

Video sequence	Accuracy (%)	Precision (%)	Recall (%)	Specificity (%)
Beauty	95	96	95	97
Flower Focus	94	98	93	96
Flower Pan	96	97	97	94
Honey bee	95	93	96	96
Jockey	95	92	97	97

Bitrate Reduction: Bitrate reduction is the percentage decrease in the bitrate of a compressed video as compared to the original uncompressed video. It represents the compression algorithms efficacy in lowering the video file's size while keeping adequate visual quality. It is determined using (17):

$$\Delta B(\%) = \left(\frac{B_c - B_o}{B_c}\right) \times 100\% \tag{17}$$

where B_c denotes the compressed bitrate of the video frame, and B_o represents the original bitrate of the video frame.

Computational time: The computational time required to encode the video input sequence using the proposed method is calculated using (18), whereas decoding time is defined as the time required to decode the compressed bitstream and recreating the video frames. Less encoding and decoding times indicates faster processing and greater computational efficiency.

$$C = \left(\frac{T_\theta - T_h}{T_h}\right) \text{x } 100\% \tag{18}$$

where, C denotes the proposed model's computational efficiency, T_θ denotes the obtained time required for encoding or decoding using the proposed approach, T_h denotes the estimated expectation of time required for encoding or decoding.

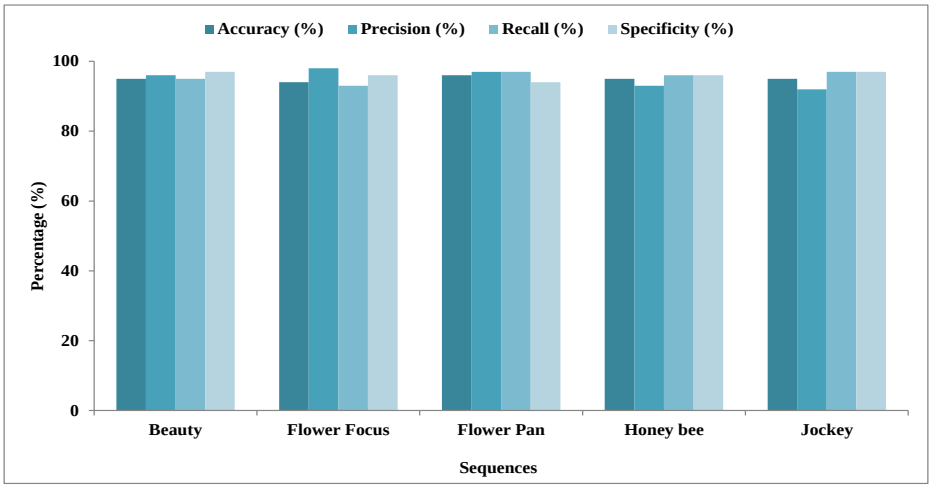


Fig. 5. Performance of the proposed model

Table 7. and Fig. 6 summarize the video sequences with their compression efficiency and processing times. Moreover, the PSNR and SSIM advantages explored through the higher in-loop filtering performance of REDNetDF and CNNSAO. This means that REDNetDF possess better artifact removal capability than the current state of the art approaches, and the results show that using REDNetDF can make significant improvement on video quality. The conventional techniques like DF and SAO provide fairly good level of quality and works fine, but the utilization of deep learning based architectures into the REDNetDF results in improved PSNR and SSIM values which reveal optimal details of videos and less compression noise. Higher bit rate reduction percentages indicate more effective compression. Longer encoder times suggest more resource intensive processes, while longer decoder times affect real time playback and quick access to decompressed video data in applications. It outlines compression efficiency and processing times for various video sequences. Notably, "Beauty" achieved a 27.52% reduction in bit rate with an encoder time of 90.323 seconds and a decoder time of 2.4 seconds. "Flower Focus" and "Honey Bee" followed with similar reductions at longer encoding and decoding times, while "Jockey" demonstrated a 25.36% reduction with moderate processing times. On average, across all sequences, there was a 24.90% reduction in bit rate, with encoding taking around 90.50 seconds and decoding 4.5 seconds, highlighting variations in efficiency and computational demands in video compression.

Table 7. Bit rate reduction and Computational time

Video sequence	Bit rate reduction	Encoder time (sec)	Decoder time (sec)
Beauty	-27.52 %	90.323	2.40
Flower Focus	-22.63 %	120.578	5.45
Flower Pan	-23.06 %	92.654	5.652
Honey Bee	-26.12 %	92.570	4.789
Jockey	-25.36 %	96.542	3.25
Average	-24.90 %	90.500	4.50

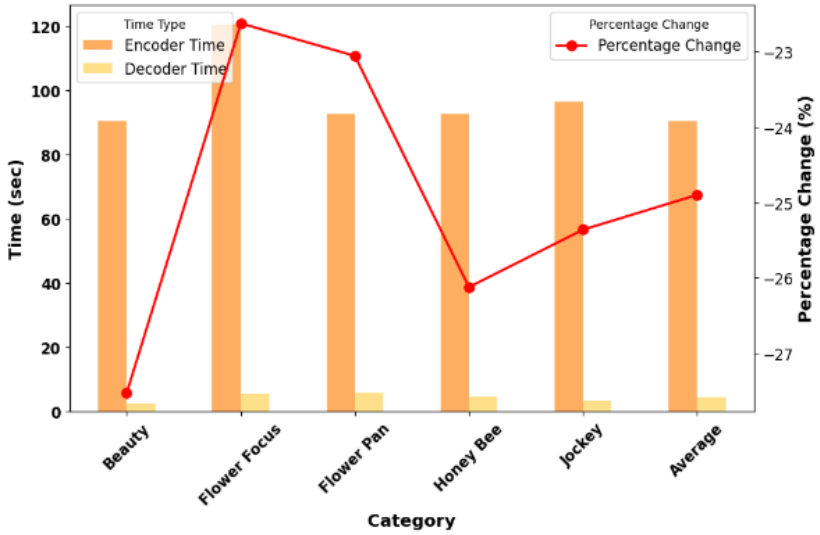


Fig. 6. Bit rate reduction and Computational time

Mean Square Error (MSE): The proposed model's MSE is calculated by averaging the squares of the errors, or the mean of the squared variance between the anticipated and observed results, which is provided in (19).

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[G(i,j) - H(i,j)]^2 \quad (19)$$

Here $G(i,j)$ *and* $H(i,j)$ represent the pixel values of pictures G and H at location (i, j) respectively. The values m and n reflect the height and breadth dimensions of images G and H, respectively.

Fig. 7 reveals the extent well estimated outputs align with actual values and highlights prediction accuracy through MSE. Blue dots represent individual bitrate data pairs. The MSE line, sloping upwards, indicates a positive correlation between estimated and obtained outputs. Variance is shown by the spread of blue dots around the MSE line, with tighter clustering suggesting better accuracy.

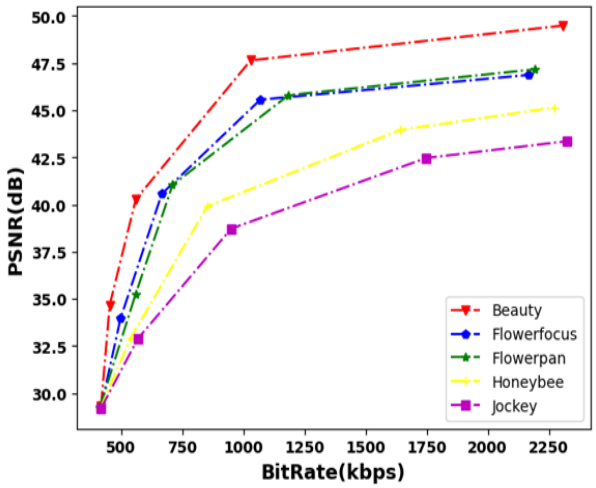


Fig. 7. Mean Squared Error Representation

Peak Signal to Noise Ratio (PSNR): The PSNR measures the difference between the original uncompressed and reconstructed video quality. Equation (20) determines the mean variance among matched pixels in the original and rebuilt frames, with higher PSNR values suggesting greater reconstruction quality.

$$PSNR = 10.\log_{10}\left(\frac{MAX_I^2}{MSE}\right) \tag{20}$$

The term MAX_I^2 refers to the maximum achievable squared pixel value. MSE is the average squared difference between matching pixels in the original and reconstructed video frames.

Fig. 8 illustrates the relationship between Bit Rate (kbps) and PSNR (dB) for different video sequences: Beauty, Flower focus, Flower pan, Honeybee, and Jockey. PSNR, a measure of video quality, improves with higher bit rates for all sequences, indicating better quality with more data. The proposed model consistently achieves the highest PSNR values, maintaining high quality at lower bit rates.

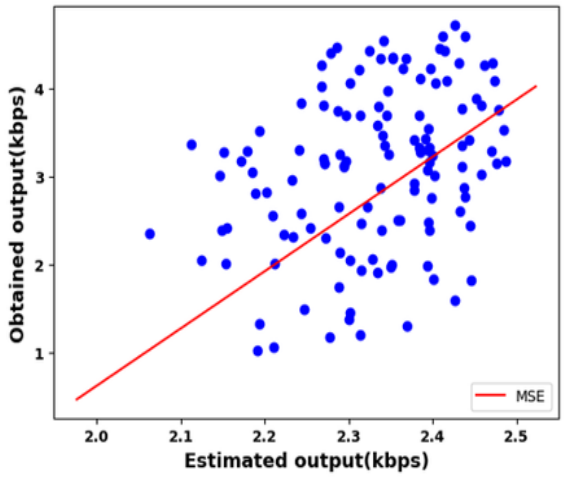


Fig. 8. PSNR vs Bitrate

Structural Similarity Index (SSIM): SSIM is a metric for comparing the structural similarity of original and reconstructed video frames. It compares the brightness, contrast, and structural similarity of corresponding image patches. Higher SSIM scores indicate greater perceptual quality and retention of structural information. It is determined using (21):

$$\mathrm{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{21}$$

where μ_x and μ_y indicate the average intensity of pixels or luminance of the raw video frame x and the compressed video frame y, respectively. σ_x^2, σ_y^2 Represent the variation measure of pixel intensities within the original and compressed video frames x and y. σ_{xy} represents the correlation between pixel intensities of the original and compressed video frames x and y. c_1 and c_2 are constants.

Fig. 9 illustrates the relationship between Bit Rate (kbps) and Structural Similarity Index (SSIM) for different video sequences (Beauty, Flower focus, Flower pan, Honeybee, and Jockey). As the bit rate increases, SSIM values improve, indicating higher video quality and better similarity to the original. Each curve shows a saturation point where further bit rate increases yield minimal SSIM gains. The proposed model achieves the highest SSIM values at lower bit rates, suggesting it is easier to compress without quality loss.

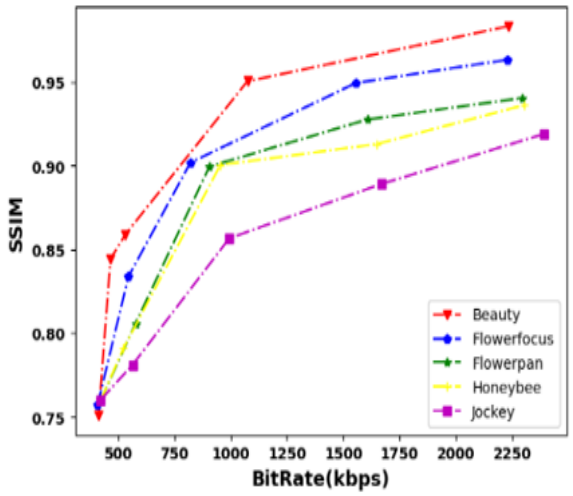


Fig. 9. SSIM Vs Bitrate

5.4. Performance Comparative Analysis

Table 8. and Fig. 10 showcase the Accuracy, Precision, Recall and Specificity of four models including the proposed model. The performance comparison demonstrates the clear advantages of the proposed REDNetDF+CNNSAO model over both traditional and recent state-of-the-art video coding techniques. While earlier models such as EDCNN-HEVC, CNN-DF-SHVC, and CA-CNN-HEVC achieved moderate accuracy and precision levels ranging from 65% to 81.67%, recent advanced models like TransVC, DRL-HEVC, and GAN-Comp have improved these metrics significantly, reaching up to 92.1% accuracy. Despite this progress, the proposed model outperforms all listed methods, achieving the highest accuracy of 95%, along with superior precision, recall, and specificity values. This indicates that the REDNetDF+CNNSAO framework not only provides more accurate and reliable detection but also reduces false positives more effectively, confirming its robustness and efficacy as a cutting-edge solution for video coding and reconstruction tasks.

Table 8. Comparative analysis of proposed model

Method	Accuracy (%)	Precision (%)	Recall (%)	Specificity (%)
EDCNN-HEVC [26]	71.33	75	65	73.33
CNN-DF-SHVC [27]	78	69	65.57	67
CA-CNN-HEVC [28]	81.67	69	72	74.67
TransVC [29]	88.5	85	84	87
DRL-HEVC [30]	90.3	88	89	90
GAN-Comp [31]	92.1	91	90.5	91.3
Proposed	95	95.2	95.6	96

This analysis highlights how the robustness of the proposed REDNetDF+CNNSAO method is superior to both traditional and state-of-the-art video coding models. It consistently delivers higher detection accuracy and is significantly less prone to false positive results. This marks it as the most effective and efficient method among all compared. Fig. 11 and Fig. 12 illustrate the bit reduction rate and computational complexity across various models including EDCNN-HEVC, CNN-DF-SHVC, CA-CNN-HEVC, and recent advanced methods such as TransVC, DRL-HEVC, and GAN-Comp. While recent models have made notable advancements, the Proposed REDNetDF+CNNSAO framework stands out by achieving the highest bit rate reduction and lowest computational complexity, making it ideal for real-time video coding and reconstruction scenarios.

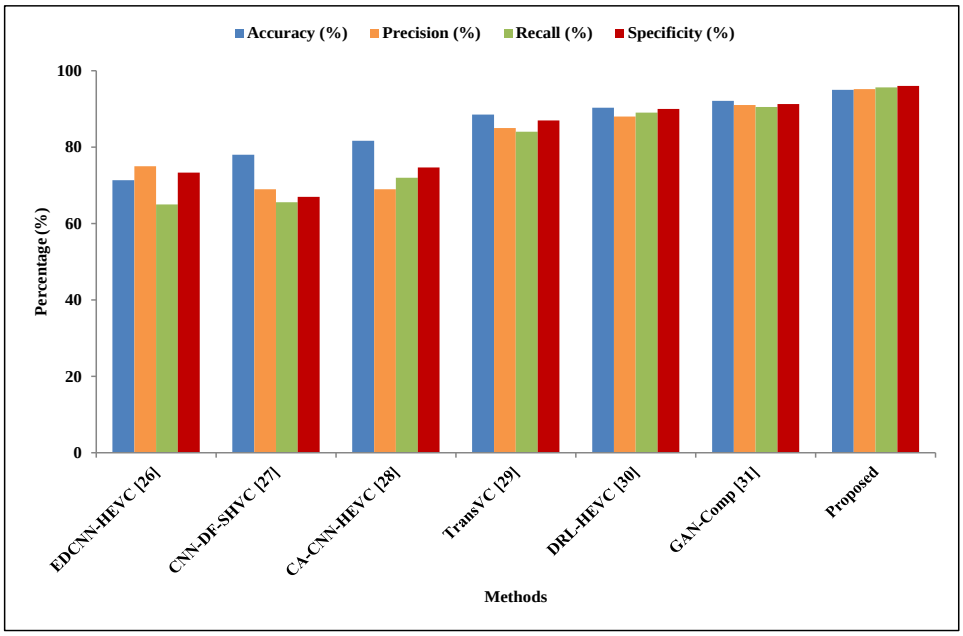


Fig. 10. Performance metric comparison

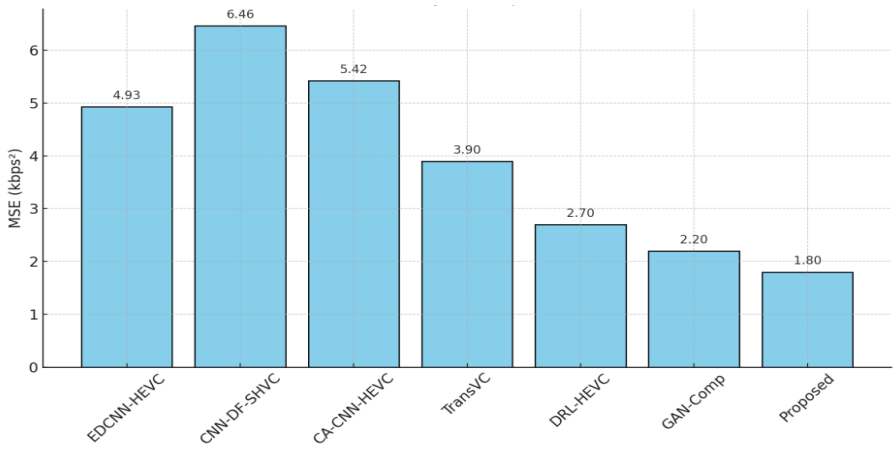


Fig. 11. Bitrate reduction

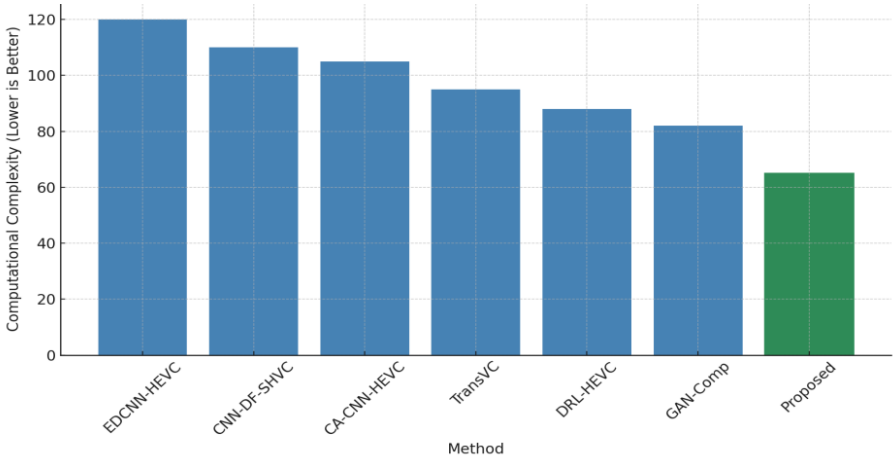


Fig. 12. Comparative analysisi of computational complexity

The performance comparison presented in the Table 9 and Fig. 13 clearly highlights the superiority of the proposed method over both traditional and recent state-of-the-art video coding techniques. Traditional models such as EDCNN-HEVC, CNN-DF-SHVC, and CA-CNN-HEVC show moderate results with PSNR values ranging from 26.9 dB to 39.8 dB and SSIM values up to 0.78, indicating limited visual quality reconstruction and structural similarity. Recently

developed models like TransVC, DRL-HEVC, and GAN-Comp exhibit noticeable improvements, with GAN-Comp achieving a PSNR of 46.2 dB and SSIM of 0.93. However, the proposed REDNetDF+CNNSAO model outperforms all others, achieving the highest PSNR of 49.7 dB, SSIM of 0.97, and the lowest MSE of 1.8 kbps ². This demonstrates its exceptional capability in preserving visual fidelity and reducing reconstruction error, making it the most robust and efficient solution among all compared methods.

Table 9. Comparative Analysis of PSNR, SSIM, MSE

Method	PSNR (dB)	SSIM	MSE ($kbps^2$)
EDCNN-HEVC [26]	26.9	0.68	4.93
CNN-DF-SHVC [27]	35.6	0.74	6.46
CA-CNN-HEVC [28]	39.8	0.78	5.42
TransVC [29]	42.3	0.85	3.9
DRL-HEVC [30]	44.5	0.90	2.7
GAN-Comp [31]	46.2	0.93	2.2
Proposed	49.7	0.97	1.8

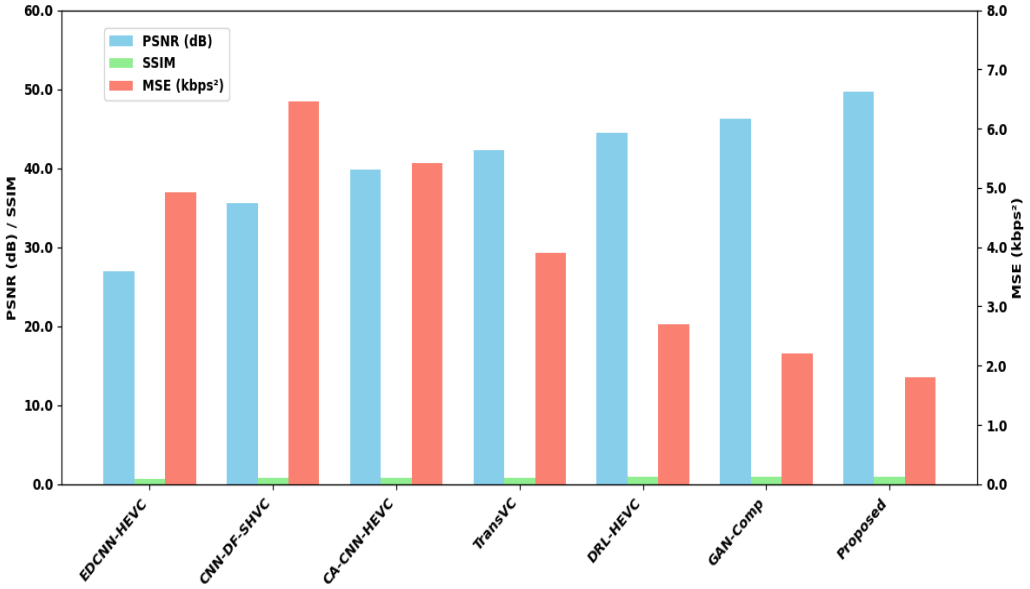


Fig. 13. Comparative of PSNR, SSIM and MSE

5.5. Ablation study

The ablation study demonstrates that REDNetDF primarily reduces blocking artifacts, significantly improving PSNR and SSIM, while CNNSAO targets ringing artifacts and enhances spatial consistency with moderate gains. When combined in the proposed hybrid framework, these filters complement each other, resulting in a substantial improvement in video quality achieving a PSNR increase of over 10 dB and a notable reduction in MSE compared to using either filter alone, mentioned in Table 10. This confirms that both components are essential and work synergistically to deliver the superior reconstruction performance reported.

Table 10. Ablation study

Method	PSNR (dB)	SSIM	MSE
REDNetDF only	39.2	0.82	4.1
CNNSAO only	38.5	0.79	4.8
Proposed REDNetDF + CNNSAO	49.7	0.97	1.8

5.6. Statistical Validation of Proposed Method on UVG Dataset

The statistical analysis of the proposed REDNetDF + CNNSAO filtering method was conducted on the UVG dataset, encompassing 16 diverse video sequences. The mean PSNR value achieved was 49.7 dB with a low standard deviation of 0.8 dB, while the SSIM averaged at 0.97 with a standard deviation of 0.02. Table 11 summarises the PSNR and SSIM. These small deviations demonstrate that the method consistently delivers superior video quality across different sequences, highlighting its robustness and reliability. The statistical validation thus reinforces the significance of the performance improvements reported in this study, confirming that the results are not due to chance or specific to individual test cases.

Table 11. Statistical Validation of Proposed Method

Metric	Mean Value	Standard Deviation (±)
PSNR (dB)	49.7	0.8
SSIM	0.97	0.02

6. Conclusion

This work has introduced an improved in-loop filtering mechanism in HEVC codec that includes the REDNetDF and CNN for implementing SAO filter. The described approach aimed to overcome the drawbacks of conventional methods and provide the improvement of the video quality, as well as the preservation of the coding efficiency. The results based on the experimental evaluation with the help of the UVG data proved the enhancement of the crucial measures. The method realized the PSNR of 49.7dB, SSIM of 0.97 and MSE of 1.8, which theoretically shows the quality of reconstruction is high, and the loss of image is minor. Moreover, the average of 24.9% reduction in the bitrate emphasizes the effectiveness of the model in loss of quality during compression of the video data. All these results are better than the existing techniques, which also clearly demonstrates the effectiveness of the proposed framework for practical use.

Acknowledgment

We would like to express our gratitude to the reviewers for their recommendations that improved the presentation.

References

[1] T. Li, M. Xu, C. Zhu, R. Yang, Z. Wang, and Z. Guan, "A deep learning approach for multi-frame in-loop filter of HEVC," *IEEE Trans. Image Process.*, vol. 28, no. 11, pp. 5663–5678, Nov. 2019.

[2] S. Kuanar, K. R. Rao, C. Conly, and N. Gorey, "Deep learning based HEVC in-loop filter and noise reduction," *Signal Process. Image Commun.*, vol. 99, Art. no. 116409, 2021.

[3] Z. Pan, X. Yi, Y. Zhang, B. Jeon, and S. Kwong, "Efficient in-loop filtering based on enhanced deep convolutional neural networks for HEVC," *IEEE Trans. Image Process.*, vol. 29, pp. 5352–5366, 2020.

[4] S. Kuanar, C. Conly, and K. R. Rao, "Deep learning based HEVC in-loop filtering for decoder quality enhancement," in *Proc. 2018 Picture Coding Symp. (PCS)*, pp. 164–168, Jun. 2018.

[5] 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.

[6] G. Raja, A. Khan, A. K. Khan, and M. H. Yousaf, "Performance analysis of HEVC in-loop filter," *Life Sci. J.*, vol. 10, pp. 331–336, 2013.

[7] I. Hautala, J. Boutellier, J. Hannuksela, and O. Silven, "Programmable low-power multicore coprocessor architecture for HEVC/H.265 in-loop filtering," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 7, pp. 1217–1230, Jul. 2014.

[8] V. Moji and M. Murugavelu, "Adaptive and efficient hybrid in-loop filter based on enhanced generative adversarial networks with sample adaptive offset filter for HEVC/H.265," *Period. Polytech. Electr. Eng. Comput. Sci.*, 2023. [Online]. Available: https://doi.org/10.3311/PPee.20881

[9] X. Meng, C. Chen, S. Zhu, and B. Zeng, "A new HEVC in-loop filter based on multi-channel long-short-term dependency residual networks," in *Proc. 2018 Data Compression Conf.*, pp. 187–196, Mar. 2018.

[10] W. S. Park and M. Kim, "CNN-based in-loop filtering for coding efficiency improvement," in *Proc. 2016 IEEE 12th IVMSP Workshop*, pp. 1–5, Jul. 2016.

[11] W. Sun, X. He, H. Chen, S. Xiong, and Y. Xu, "A nonlocal HEVC in-loop filter using CNN-based compression noise estimation," *Appl. Intell.*, vol. 52, no. 15, pp. 17810–17828, 2022.

[12] X. Lu, C. Yu, and X. Jin, "A fast HEVC intra-coding algorithm based on texture homogeneity and spatio-temporal correlation," *EURASIP J. Adv. Signal Process.*, vol. 2018, Art. no. 1, 2018.

[13] Z. Pan, X. Yi, Y. Zhang, B. Jeon, and S. Kwong, "Efficient in-loop filtering based on enhanced deep convolutional neural networks for HEVC," *IEEE Trans. Image Process.*, vol. 29, pp. 5352–5366, 2020.

[14] Q. Xu, X. Jiang, T. Sun, and A. C. Kot, "Detection of transcoded HEVC videos based on in-loop filtering and PU partitioning analyses," *Signal Process. Image Commun.*, vol. 92, Art. no. 116109, 2021.

[15] A. Dhanalakshmi and G. Nagarajan, "Combined spatial temporal-based in-loop filter for scalable extension of HEVC," *ICT Express*, vol. 6, no. 4, pp. 306–311, 2020.

[16] P. R. Christopher and S. Sathasivam, "Quality assessment of dual-parallel edge deblocking filter architecture for HEVC/H.265," *Appl. Sci.*, vol. 12, no. 24, Art. no. 12952, 2022.

[17] S. Y. Lim, M. K. Choi, and Y. L. Lee, "Frequency-based adaptive interpolation filter in intra prediction," *Appl. Sci.*, vol. 13, no. 3, Art. no. 1475, 2023.

[18] T. Mallikarachchi, D. Talagala, H. Kodikara Arachchi, C. Hewage, and A. Fernando, "A decoding-complexity and rate-controlled video-coding algorithm for HEVC," *Future Internet*, vol. 12, no. 7, Art. no. 120, 2020.

[19] M. Li and W. Ji, "Screen content-aware video coding through non-local model embedded with intra-inter in-loop filtering," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 35, no. 2, pp. 1870–1883, Feb. 2025. [Online]. Available: https://doi.org/10.1109/TCSVT.2024.3473543

[20] F. Bossen, B. Bross, K. Suhring, and D. Flynn, "HEVC complexity and implementation analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1685–1696, Dec. 2012.
[21] M. Zhou, W. Gao, M. Jiang, and H. Yu, "HEVC lossless coding and improvements," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1839–1843, Dec. 2012.
[22] G. J. Sullivan and J. R. Ohm, "Recent developments in standardization of high-efficiency video coding (HEVC)," in *Proc. Appl. Digit. Image Process. XXXIII*, vol. 7798, pp. 239–245, 2010.
[23] I. K. Kim, J. Min, T. Lee, W. J. Han, and J. Park, "Block partitioning structure in the HEVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1697–1706, Dec. 2012.
[24] J. L. Lin, Y. W. Chen, Y. W. Huang, and S. M. Lei, "Motion vector coding in the HEVC standard," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 6, pp. 957–968, Dec. 2013.
[25] A. Mercat, M. Viitanen, and J. Vanne, "UVG dataset: 50/120fps 4K sequences for video codec analysis and development," in *Proc. 11th ACM Multimedia Syst. Conf.*, pp. 297–302, May 2020. [Online]. Available: https://ultravideo.fi/dataset.html and Available: https://doi.org/10.1145/3339825.3394937
[26] Z. Pan, X. Yi, Y. Zhang, B. Jeon, and S. Kwong, "Efficient in-loop filtering based on enhanced deep convolutional neural networks for HEVC," *IEEE Trans. Image Process.*, vol. 29, pp. 5352–5366, 2020.
[27] A. Dhanalakshmi and G. Nagarajan, "Convolutional neural network-based deblocking filter for SHVC in H.265," *Signal Image Video Process.*, vol. 14, no. 8, pp. 1635–1645, 2020.
[28] 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.
[29] Y. Li, L. Li, Z. Zhuang, Y. Fang, H. Peng, and N. Ling, "Transformer‐based data‐driven video coding acceleration for industrial applications," *Math. Probl. Eng.*, vol. 2022, Art. no. 1440323, 2022.
[30] N. Li, Y. Zhang, L. Zhu, W. Luo, and S. Kwong, "Reinforcement learning based coding unit early termination algorithm for high efficiency video coding," *J. Vis. Commun. Image Represent.*, vol. 60, pp. 276–286, 2019.
[31] P. Du, Y. Liu, N. Ling, L. Liu, Y. Ren, and M. K. Hsu, "A generative adversarial network for video compression," in *Proc. SPIE Big Data IV: Learning, Analytics, and Applications*, vol. 12097, pp. 129–136, 2022.

Authors' Profiles

Vanishree Moji received her B.E. degree in Electronics and Communication Engineering from Shri Taralabalu Jagadguru Institute of Technology (STJIT), Ranebennur of Karnataka University, Dharawad and M. Tech. degree in Digital Electronics and Communication Systems from PES Institute of Technology, Bengaluru of Visvesvaraya Technological University, Belagavi. She has 15 years of teaching experience in various Engineering colleges. She is currently working towards her Research Work under the Department of Electronics and Communication Engineering, K S Institute of Technology, Bengaluru of Visvesvaraya Technological University, Belagavi. Her research interests include Digital Image Processing, Video Processing, Wireless Communication Systems, Electronics and Communication Engineering. She holds life membership of ISTE.

Bharathi Gururaj received her Bachelor's degree from the PES Institute of Technology, Bangalore University, M. Tech from M S Ramaiah Institute of Technology, Bangalore, and Ph.D. degree from the VTU, Belagavi. She was employed as Assistant Professor in SVCE, Bengaluru. She also worked as an Associate Professor and Head of Department of Electronics and Communication Engineering at ACS College of Engineering, Bengaluru. She is currently working as Associate Professor at Department of Electronics and Communication Engineering, K S Institute of Technology, Bengaluru. Her research interests are in the Image and Video Processing, Wireless Communication System. She has contributed in more than 10 research journals. Her research papers were published extensively in leading international journals and conferences. She has reviewed around 50+ papers for various Journals, IEEE Conferences and International Journals. She is an Editorial Board member and Reviewer for Science Publishing Group Journal.

Mathivanan Murugavelu obtained his B.E degree in ECE from University of Madras (2001), M.E degree in Applied Electronics (2007) and Ph.D in ECE from Anna University, Chennai (2014). He has more than 20 years of teaching experience and more than 10 years of research experience. He has published more than 30 research articles including SCI, Scopus indexed International Journals, Book chapters and National & International Conferences. His area of interest includes Signal Processing, Speech & Image Processing, Networking and IoT. Presently he is working as Professor, Department of ECE, Salem College of Engineering & Technology, Salem, Tamilnadu affiliated to Anna University. He has reviewed many research papers in various Journals and Conferences. He has acted as Indian Examiner for the research scholars of few Universities. He is the life time member of ISTE and IETE professional societies. He is guiding the research for 4 candidates and 2 of them completed their Ph.D under VTU, Karnataka.

How to cite this paper: Vanishree Moji, Bharathi Gururaj, Mathivanan Murugavelu, "Enhancing In-loop Filter of HEVC with Integrated Residual Encoder-Decoder Network and Convolutional Neural Network", International Journal of Image, Graphics and Signal Processing(IJIGSP), Vol.17, No.4, pp. 49-67, 2025. DOI:10.5815/ijigsp.2025.04.04