[1]Chena Ram
[2]Subhash Panwar

High-Speed CTU-Based HEVC Intra-Coding via Deep Learning Technique

Journal of Electrical Systems

Abstract: - In recent years, High-Efficiency Video Coding (HEVC) standards have gained widespread popularity owing to their ability to provide improved video compression performance. Nevertheless, the HEVC intra-coding procedure is still computationally demanding, which prevents its real-time application on devices with limited resources. In order to overcome this difficulty, we provide a deep learning-based method for quick CTU-based intra-coding in HEVC that makes use of a unique Convolutional Neural Network (CNN) to forecast intra-coding modes. Our method achieves an average encoding time reduction of 68.69% on the RAISE dataset and 62.26% on the JCT-VC test set, with a BD-BR increase of 1.238% and BD-PSNR loss of -0.084 dB, outperforming LeNet-5 and OSDN. This advancement enhances video coding efficiency for real-time applications.

Keywords: Convolutional Neural network, CTU partitioning, Deep learning, High Efficiency Video Coding, Intra-coding.

I. INTRODUCTION

The HEVC standard has significantly advanced video compression by offering superior coding efficiency compared to its predecessor, the H.264/AVC standard. However, the intra-coding process in HEVC remains computationally intensive, which hinders its real-time implementation on various devices. To address this challenge, this manuscript presents a novel approach that leverages deep learning to accelerate the CTU-based intra-coding process in HEVC. By reducing computational complexity while maintaining coding efficiency, our proposed method aims to enable real-time implementation of HEVC on diverse devices. Unlike prior works [30, 34], our method introduces a multi-branch CNN with early termination logic, reducing unnecessary computations by up to 72.41% at QP=37. With Versatile Video Coding (VVC) emerging as a new standard, we discuss its potential integration in Section 6, enhancing our method's relevance.

Achieving excellent video quality for transmission and storage is a fundamental requirement in video encoding. In today's era of rapid Internet technology development, the usage of media data, such as images, audio, and videos, has become increasingly common and prevalent in people's daily lives [1]. Rapid advancements in technology have led to a significant increase in the popularity and adoption of flat-panel TV sets, with recent predictions [2] indicating that two-thirds (66 percent) of installed flat-panel TV sets will be Ultra-High-Definition (UHD) by 2023, a significant rise from 33 percent in 2018. The advent of Ultra-High-Definition (UHD), or 4K, video streaming is one of the main causes of this increase. However, because of its significant impact on network traffic, this technical innovation presents new issues.

The standards for UHD video quality are substantially higher than those for high-definition (HD) video. This is mainly because 4K video typically has a bit rate of 15 to 18 Mbps, which is nine times greater than Standard-Definition (SD) video and more than double that of HD video [2]. Video service providers are now confronted with the critical challenge of satisfying end consumers' increasing demands for video quality.

In order to attend online classes during the COVID-19 pandemic, students mostly depended on video-conferencing solutions. However, high-speed video streaming is necessary for a continuous viewing experience. Because of this, it is now more important than ever to eliminate video-encoding delays in the most recent video coding standards. Researchers have focused on creating effective coding methods in order to solve these issues and enhance the video compression process. Specifically, the HEVC standard outperforms its predecessors in terms of compression performance. HEVC uses a variety of coding tools and methods to accomplish this, such as the extremely adaptable Coding Tree Unit (CTU) structure for intra-coding.

Nevertheless, there is still space for development in terms of raising the effectiveness of CTU-based intra-coding, even with the advancements in HEVC. Using deep learning algorithms is a promising way to address this problem. By automatically learning hierarchical representations from massive amounts of data, deep learning has demonstrated impressive performance in a number of fields, including computer vision and natural language

[1]*Corresponding author: Dept. of ECE, Engineering College Bikaner, India
[2]Dept. of Information Technology, Engineering College Bikaner, India
Copyright©JES2025 on-line: journal.esrgroups.org

processing [3]. The intrinsic spatial and temporal relationships found in video sequences can be exploited by adding deep learning techniques to the CTU-based intra-coding process, improving coding efficiency.

A. Contributions

This study makes the following key contributions to the field of HEVC intra-coding:

1) We propose a novel CNN architecture tailored for fast CTU-based intra-coding, reducing encoding time by an average of 68.69% on the RAISE dataset and 62.26% on the JCT-VC test set, with negligible rate-distortion (RD) loss.

2) We introduce an early termination mechanism within the CNN that bypasses unnecessary splitting decisions, enhancing computational efficiency across varying quantization parameters (QPs).

3) We provide a comprehensive evaluation against state-of-the-art methods (LeNet-5 and OSDN), demonstrating superior encoding time savings and RD performance across diverse resolutions and video sequences.

4) We leverage a diverse dataset (RAISE) and the JCT-VC test set to ensure robustness and generalizability, addressing practical video compression needs.

Together, these contributions make it possible to implement HEVC in real-time on devices with limited resources.

The structure of the manuscript is as follows: Background and related work are reviewed in Section II; CTU partitioning and intra-prediction are described in Section III; our suggested approach is presented in Section IV; experimental results are reported in Section V; and future approaches are discussed in Section VI.

II. Background and Related Work

Prior studies have concentrated on several methods to improve HEVC intra-coding coding efficiency and lower its computational cost. These technologies include parallel processing techniques, early termination strategies, and mode decision algorithms. Nonetheless, there is a growing interest in using deep learning methods to successfully handle these issues.

In contemporary video coding standards like H.264 and HEVC, the complexity of video encoders is far greater than that of decoders. The inclusion of multiple coding modes and the assignment of distinct modes to each block are the causes of this increased complexity. The encoding method is computationally intensive since encoders usually compare several potential modes and choose the best one for each block [4]. In order to find better coding modes, practical encoders have begun utilizing machine learning methods, especially deep learning, and heuristic algorithms.

Liu et al. [5] proposed a hardware solution for an intra-HEVC encoder that utilizes a trained CNN to assist in determining the optimal mode for coding unit (CU) partitioning. In intra prediction, a CTU is recursively divided into smaller coding units (CUs) to create a quadtree structure. The trained CNN can effectively decide whether to further divide a CU based on its content and the provided quantization parameter. In a previous study [6], various HEVC domain features were investigated for their correlation with the CU partition. Subsequently, a novel approach was introduced using a joint Support Vector Machine (SVM) classifier to leverage these features effectively in determining CU depths. The proposed method aims to reduce the encoding complexity of HEVC by bypassing the need for brute-force RDO (Rate-Distortion Optimization) search. Jin et al. [7] examined the CU splitting mode choice using Versatile Video Coding (VVC). VVC utilizes a QuadTree with nested Multi-Type Tree (QTMTT) structure, incorporating QuadTree (QT), Binary Tree (BT), and Ternary Tree (TT) partitioning, which is more complex than HEVC's quadtree structure. They trained a CNN to classify 32×32 CUs into five different ways, where each classification represents a distinct tree level.

For AVC to HEVC transcoding, Xu et al. [8] investigated the CU splitting mode selection. They proposed a hierarchical LSTM network that utilizes characteristics derived from H.264 coded bits to predict the optimal CU partitioning mode. In a similar vein, Song et al. [9] explored a CNN-based approach for HEVC intra encoder intra-prediction mode selection. They employed a CNN to generate a candidate list of the most probable modes for each 8×8 or 4×4 block based on its content and quantization parameter. Then, they used a regular rate-distortion optimization method to select the best mode from the candidate list.

Recently, learning-based techniques for reducing complexity in HEVC have emerged. Instead of performing exhaustive rate-distortion optimization searches, these techniques employ machine learning algorithms trained on large datasets to improve video encoding guidelines. Q. Hu et al. [10] described the coding unit partitioning process as a binary classification problem using logistic regression, while [11] proposed the use of Support

Vector Machines (SVM) for classification in the CU splitting procedure. As a result, well-trained classification models can be used to significantly reduce the computational time required for CU partitioning.

In [12], SVM was employed to make quick partitioning decisions, resulting in an average encoding delay reduction of 37 percent (up to 71 percent) with a maximum rate-distortion cost of 6%. However, the effectiveness of such systems heavily relies on the chosen training corpus, which requires a substantial amount of time to complete beforehand. Another notable model is the convolutional neural network [13], inspired by the visual processing in animal brains. Gradient-based learning methods [14] can be used to train CNNs with suitable architectures to recognize visual patterns in various tasks, such as handwriting recognition [15], image classification [16], and image segmentation [17].

To accelerate the original intra-coding process, [18] proposed a two-stage prediction unit (PU) size division. Redundant prediction units are filtered out based on the texture complexity of the down-sampled CTU and its sub coding units. Furthermore, [19] suggested an early stopping criterion for CU partitioning based on texture complexity. The decision to split or not split a CU, along with the selection threshold, is determined by the texture complexity of the current CU. In their work, Pan et al. [20] presented a novel approach for early quad-tree CU depth zero decision and a CU depth three skipped choice strategy, utilizing deep learning to exploit the correlation between spatially and temporally neighboring CTUs. The proposed method also takes into account the information pertaining to PU mode and optimal CTU depth selection. In [21], a fast CU partitioning and pruning method is proposed using RDO (Rate-Distortion Optimization) and Hadamard cost. The average BD-rate loss is 1.05 percent, resulting in a 38 percent reduction in encoding time.

A. Deep Learning Approaches

Deep learning techniques have garnered significant attention in a range of computer vision tasks, such as image classification, object detection, and semantic segmentation. Researchers have focused on investigating the possibilities of deep learning algorithms in video coding in recent years. These techniques seek to increase overall coding efficiency by addressing the computational complexity related to video coding. This section covers different deep learning techniques used for CTU-based intra-coding.

1) CNN-based Approaches: Convolutional Neural Networks (CNNs) are widely used in video coding, especially for tasks like mode choice and intra-prediction. By utilizing the hierarchical and local spatial connections found in video frames, these methods improve prediction accuracy while lowering coding artifacts. The use of CNNs for CTU-based intra-coding has been the subject of numerous studies, with an emphasis on increasing coding efficacy and efficiency. For instance, Liu et al. [5] proposed a CNN-based method for CTU-based intra-prediction, which demonstrated considerable improvements in coding efficiency. Their network was trained to learn spatial correlations within a CTU, leading to accurate predictions that resulted in reduced residual information and improved compression performance.

2) RNNs and LSTM-based Approaches: Recurrent Neural Networks (RNNs), specifically Long Short-Term Memory (LSTM) networks, have also been employed in video coding research. These models excel at capturing the temporal dependencies inherent in video sequences, thereby enabling more accurate predictions and improved compression efficiency. Bouaafia et al. [26] introduced an LSTM-based approach for CTU-based inter-prediction. Their network was trained to learn the temporal dynamics and dependencies present in video frames. By effectively exploiting sequential information, the LSTM model achieved superior coding performance compared to traditional methods.

3) GAN-based Approaches: Generative Adversarial Networks (GANs) have demonstrated promising results across various computer vision tasks, such as image synthesis and super-resolution. In the context of video coding, GANs have been employed to enhance the quality of reconstructed frames and mitigate coding artifacts. Linwei et al. [27] proposed a GAN-based approach for CTU-based intra-coding. In their work, the generator network learned to generate high-quality intra-predicted frames, while the discriminator network distinguished between real and generated frames. Through adversarial training, the proposed approach improved the visual quality and coding efficiency of the reconstructed frames.

B. Comparetive Review

The increasing complexity of implementing HEVC has led researchers to explore machine learning and deep learning approaches as potential solutions. In recent years, significant progress has been made in developing fast learning-based partition methods to improve the efficiency and coding performance of CU partitioning. In this

section, we present a comprehensive review of recent works that have employed machine learning and deep learning techniques to tackle the challenges posed by the high complexity of HEVC coding.

TABLE I. COMPARATIVE REVIEW OF PAST WORK

Study	Year	Approach	Key Contributions	Encoding Time Saving	Limitations
Momcilovic et al. [4]	2015	Runtime machine learning approach	Efficient CU splitting decisions without pre-training	Approximately 47.5%	Modest improvement, limited testing
Liu et al. [11]	2016	Support Vector Machines (SVM)	CU size determination based on image attributes	52.99%	Manual feature extraction
Liu et al. [5]	2016	Convolutional Neural Networks (CNNs)	Fast CNN-based algorithm for CU partitioning	Approximately 60.7%	Shallow CNN architecture, limited training data
Feng et al. [28]	2018	Texture features and CNN classification	Fast CTU depth decision algorithm	27.54%	Cost calculation within depth range required
Xu et al. [29]	2018	Hierarchical CNN and LSTM network	Intra-mode and inter-mode CU partitioning	Approximately 64.5%	Shallow network depth
Ting et al. [30]	2019	CNNs for intra-frame mode decision	Reduced computational complexity	Up to 66.59%	LeNet-5 model limitations
Kim et al. [31]	2019	Convolutional network model	Quick CTU depth prediction and skipping RDO process	61.77%	Impact on encoding quality
Chen et al. [32]	2020	Shallow asymmetric-kernel CNN (AK-CNN)	CU split and intra-mode choice with low complexity	75.2%	Four classifiers required
Zaki et al. [33]	2021	CtuNet with ResNet18 CNNs	Quick CTU splitting in HEVC intra coding	63.68%	Evaluation lacked video quality consideration
Wang et al. [34]	2021	One-Stage Decision Network (OSDN) model	Predicting CU partitioning and 35 intra-frame prediction modes	73.69%	Long training time, room for improvement in quality

Table I provides an overview of notable methods employed in recent years. Momcilovic et al. [4] proposed a runtime machine learning approach for efficient Coding Unit (CU) splitting decisions, which dynamically adjusts to changes in video content without the need for pre-training. However, concerns arise due to its relatively modest improvement in encoding time delay (approximately 47.5%) and the limited testing conducted on only nine sequences, raising questions about the overall performance of the model.

Liu et al. [11] proposed an efficient CU size determination algorithm based on Support Vector Machines (SVM). This approach categorizes each CU as homogeneous, complex, or uncertain by extracting valuable image attributes. Through this approach, the algorithm can efficiently skip complex Coding Units (CUs) while also terminating homogeneous CUs early, thereby optimizing the intra coding process. The results demonstrate a significant encoding time saving of 52.99% for HEVC intra coding. However, this method had limitations due to the need for manual feature extraction. In contrast, convolutional neural networks (CNNs) can automatically extract relevant features from the original image and efficiently predict CU partitioning.

Liu et al. developed a fast CNN-based algorithm that minimizes the CU partition modes in each CTU prior to rate-distortion optimization (RDO) processing, effectively reducing the encoder's complexity [5]. They also developed a high-speed and low-cost accelerator for fast algorithm implementation. The encoding delay was improved by approximately 60.7% for intra coding. However, the limited training data resulted in a shallow CNN architecture, leading to increased computational cost when learned model needs to be used multiple times.

To address the computational burden associated with conventional methods, we draw inspiration from Feng et al. [28], who introduced a fast CTU depth decision algorithm by incorporating texture features and employing CNN classification technology. The key advantage of Feng et al.'s algorithm lies in its ability to efficiently skip the rate-distortion cost computations outside the predicted depth range, thus significantly reducing computational overhead. However, despite this improvement, their method still necessitates the calculation of costs within the divided depth range. Furthermore, Xu et al. introduced a hierarchical CNN structure for intra-mode CU partition prediction and incorporated a long- and short-term memory (LSTM) network for inter-mode CU partitioning, exploiting time correlation in CU partitioning [29]. Although this approach reduced coding complexity, its shallow network depth limited its capability to handle complex CU partitioning.

Ting et al. introduced a pioneering approach by considering it as a classification task and effectively mitigated the computational complexity of rate-distortion optimization (RDO) through the use of Convolutional Neural Networks (CNNs) [30]. While this approach was promising, the adopted LeNet-5 model structure was not capable of capturing more complex features. Kim et al. [31] proposed a novel convolutional network model that exhibits superior performance in predicting the current CTU depth swiftly, thereby eliminating the need for the complex RDO (Rate-Distortion Optimization) process. While their approach significantly reduced the encoding

time, it also raised concerns regarding its potential impact on the encoding quality. Chen et al. [32] introduced a shallow asymmetric-kernel CNN (AK-CNN) with low complexity for both CU split and intra-mode choice, demonstrating a substantial complexity reduction of 75.2% with a slight increase in bit rate (2.09%). However, this approach required four classifiers (CNNs) to determine the Coding Unit/Partition Unit (CU/PU) size at each level of the selected Quantization Parameter (QP).

More recently, Zaki et al. [33] proposed CtuNet, a deep learning-based system for quick CTU splitting in HEVC intra coding. The system employed three CNN models based on the ResNet18 architecture, trained from scratch using the CtuNet framework and the Raw Image Dataset (RAISE), which encompassed four alternative resolutions. The results demonstrated a promising encoding time saving of 63.68%. However, it should be noted that the evaluation did not consider video quality degradation, and the use of a pre-trained model based on object detection may not be suitable for this specific problem. Finally, Wang et al. proposed the One-Stage Decision Network (OSDN) model, which predicted CU partitioning results and 35 different intra-frame prediction modes, achieving greater coding time savings compared to previous works [34]. Nevertheless, the model's training period was unduly lengthy, and the encoding quality could still be improved.

Among these techniques, machine learning-based solutions had issues reliably capturing the features required for model training and relied on human feature extraction. CNN-based network models, like the one suggested in this study, on the other hand, performed better in terms of reducing encoding complexity and improving accuracy. The examined strategies demonstrate how deep learning techniques outperform traditional strategies in lowering HEVC's computational complexity. The goal of ongoing research is to improve HEVC encoder performance even more by employing cutting-edge methods. In conclusion, deep learning techniques have proven to be more effective than traditional methods at lowering the computational cost of HEVC encoding. With the main goal of reducing encoding time while guaranteeing minimal impact on video quality and compression efficiency, researchers are always working to optimize the HEVC encoder's performance.

III. CTU PARTITIONING FUNDAMENTALS

In terms of lower bit-rate, the HEVC has demonstrated better performance than the earlier H.264/AVC coding standard [22–24]. However, encoding latency has significantly increased as a result of HEVC's intricate coding unit selection procedure. A hybrid coding strategy, which combines many coding techniques, is frequently used in the field of video coding systems. In particular, coding processes in HEVC occur at the coding unit (CU) level, which is established by intra-picture prediction. This procedure is essential to the effective representation and compression of video data.

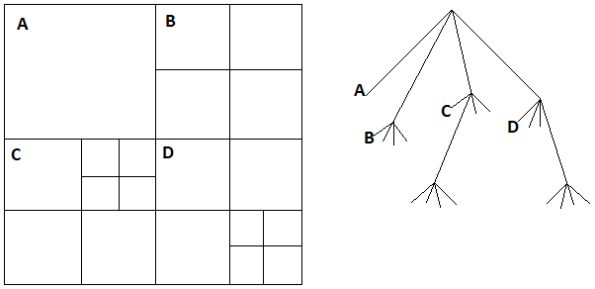

Fig. 1 A CTU partitioning and its corresponding quad tree representation

A. CTU Partitioning Scheme

In order to elucidate the CTU partitioning scheme, we refer to Figure 1. This illustrative diagram showcases an input raw video sequence that has been divided into individual images, and subsequently, each image is further partitioned into coding units (CUs). It is essential to note that within the HEVC framework [22], the CTU represents the largest block size, which can be further subdivided into smaller CUs to enhance the coding efficiency. The coding process follows a predetermined order, with earlier coded units serving as references to predict subsequent coding units. The predicted residues undergo transformation, quantization, and entropy encoding to generate the output encoded bit streams.

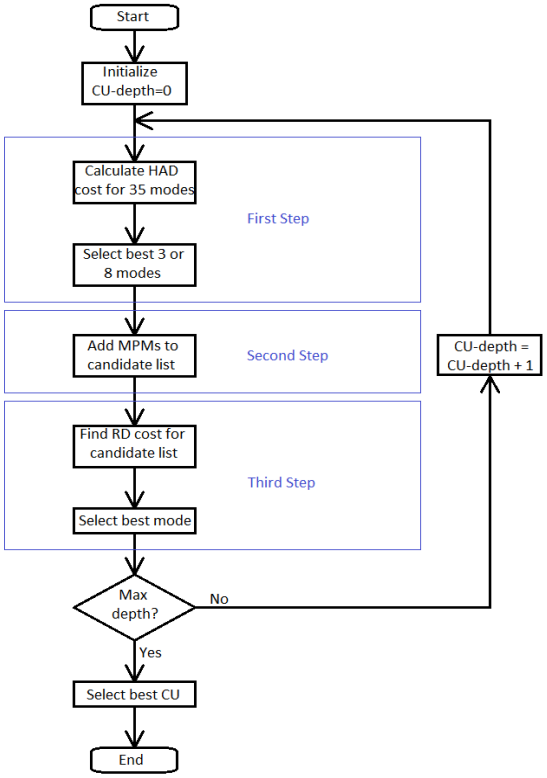

Fig. 2 HEVC intra-prediction algorithm

HEVC employs a quad-tree structure for its coding tree block, allowing for quad tree depth choices ranging from 0 to 3. The coding units at levels D0 to D3 (i.e., 64×64, 32×32, 16×16, and 8×8 pixels) are used. The number of potential segments available at each depth level is a factor of 4^i. Therefore, the total number of alternatives for coding unit selection is 1 + 4 + 16 + 64 = 85 distinct CU allocations. HEVC determines the CU sizes in each CTU through a brute-force rate-distortion optimization search [25]. This search involves a top-down inspection procedure and a bottom-up comparison procedure. Notably, our method leverages deep learning to optimize the recursive quad-tree CTU splitting process, which significantly contributes to the encoding time in HEVC [22].

B. Intra-Picture Prediction

Intra-picture prediction involves predicting blocks within a single image. The main objective is to select the best prediction mode for each block and transmit it to the decoder. This process involves comparing the bit rate and distortion of several modes, ultimately selecting the mode with the lowest rate-distortion cost. To achieve the best possible Coding Unit (CU) split with the lowest RD cost, the HEVC encoder employs a three-step intra-prediction technique, as illustrated in Figure 2.

Step I: Initially, for each CU, the HEVC utilizes thirty-five intra-prediction modes. These modes consist of DC prediction, representing a flat surface with a value corresponding to the mean of the boundary samples; planar prediction, which assumes an amplitude surface with vertical and horizontal slopes derived from the boundaries; and 33 angular modes [22]. To evaluate the cost of each mode at a given depth, the Hadamard cost is calculated using the following formula:

$$C_{HAD}=D_{SATD}+\lambda \cdot R_m \quad (1)$$

Here, D_{SATD} represents the sum of absolute transform differences, R_m is the number of bits for the selected intra-mode m, and λ is the Lagrange multiplier determined by the quantization parameter (QP). Depending on the CU depth (zero, one, two, or three), the top three or eight intra-modes with the lowest Hadamard cost are chosen.

Step II: The most probable modes (MPMs) identified in the previous step are added to the list of potential modes that may be utilized.

Step III: During rate-distortion optimization, the mode with the lowest rate-distortion cost is selected as one of the optimal modes for the specified depth level. The rate distortion cost is calculated as follows:

$$C_{RD}=D_{SSE}+\lambda \cdot R_t \quad (2)$$

Here R_t represents the total number of bits required to encode the coding unit at the chosen depth level, and $D_{SSE}$is the distortion measure, computed using the sum of squared errors.

This process is repeated at each depth level until depth 3 is reached. The optimal coding unit partition for a given CTU, which yields the lowest rate-distortion cost, is determined through a bottom-up rate distortion cost comparison. By following this three-step process, the HEVC encoder can predict intra-picture blocks, selecting the most suitable prediction modes and achieving optimal CU splits with minimal rate-distortion costs. The need to perform such searches explains the increased computational costs of HEVC compared to its predecessor and emphasizes the importance of reducing such searches. The utilization of deep learning techniques in this context could potentially enhance the prediction process further, leading to faster CTU-based intra-coding for HEVC.

IV. PROPOSED METHOD

In this section, we describe the dataset used for training our proposed method and outline the test methodology employed to evaluate its performance. Additionally, we provide details about the software and configuration settings utilized in our experiments. Our proposed method involves training a deep neural network to predict the optimal intra-coding mode for each CTU in HEVC. We design suitable network architecture and train it on a large dataset of CTU samples, considering different content types and complexity levels. The trained model can then accurately predict the intra-coding mode for unseen CTUs, significantly reducing the computational burden.

Algorithm 1: CNN-Based Fast CTU Intra-Coding Prediction

Input: Dataset D, Config File, $QP \in \{22, 27, 32, 37\}$, CTU (Size = 64×64)

Output: Predicted Intra-Coding Modes

for each CTU in D do
// **Step 1:** Preprocessing
Extract Y_{CTU} from CTU
for each 16×16 block $B_{i,j}$ in Y_{CTU} do
Compute mean $\mu_{i,j} = \text{mean}(B_{i,j})$
$B'_{i,j} = B_{i,j} - \mu_{i,j}$
end for
$Y'_{CTU} = \{B'_{i,j}\}$
// **Step 2:** CNN Prediction
F_1 = ReLU(Conv(Y'_{CTU}, 8 kernels, 4×4))
F_2 = ReLU(Conv(F_1, 16 kernels, 4×4))
F_3 = ReLU(Conv(F_2, 32 kernels, 2×2))
V = Flatten(F_1) + Flatten(F_2) +Flatten(F_3)
$\tilde{V}$ = Concat(V, QP)
for level l = 1 to 3 do
$H_l^{(1)}$ = ReLU(FC($\tilde{V}$))
$H_l^{(2)}$ = ReLU(FC($H_l^{(1)}$))
S_l = Sigmoid(FC($H_l^{(2)}$))
D_l = ($S_l \geq 0.5$) ? 1 : 0
if l = 1 and D_1 = 0 then
$D_2 = D_3 = 0$
break
else if l = 2 and D_2 = 0 then
D_3 = 0
break
end if
end for
// **Step 3:** HEVC Integration
Replace RDO in HM 16.12 with $\{D_1, D_2, D_3\}$
Encode CTU using parameters from table II
end for
return Predicted Modes

To conduct the experiments, the HM 16.12 reference software was utilized to process the input raw images in YUV format. The HM 16.12 reference software, introduced by the JCT-VC in June 2016 [35], has been widely accepted for HEVC. In order to implement the code and establish the encoder project, the utilization of Microsoft

Visual Studio 9.0 is necessary. The successful build of the project generates two essential program files, namely encode.exe and decode.exe. To obtain the final results, execution of these two files, accompanied by the appropriate configuration settings, is required. This framework ensures an efficient and accurate evaluation of the proposed deep learning approach in the context of HEVC. The algorithm1 highlights the steps used in our proposed method using CNN-based approach.

The proposed method takes as input a raw image dataset D, consisting of images in the YUV color space, along with a configuration file (e.g., encoder_intra_main.cfg) that specifies the encoding parameters. The quantization parameter is selected from the set {22, 27, 32, 37}, controlling the compression quality, while the Coding Tree Unit size is fixed at 64×64 pixels. The output is the predicted intra-coding mode, represented as binary splitting decisions for each CTU at multiple depth levels. The algorithm leverages a CNN to predict intra-coding modes for CTUs in HEVC, replacing the computationally expensive RDO process in the HM 16.12 reference software. The method preprocesses CTUs, applies CNN-based prediction, and integrates decisions into the HEVC encoder.

Step 1: Preprocessing

For each CTU in dataset D:

1) Extract the luminance (Y) channel, denoted $Y_{CTU} \in \mathbb{R}^{64\times64}$.
2) PartitionY_{CTU} into non-overlapping 16×16 blocks, yielding $B_{i,j}$, where$i, j \in \{0, 1, 2, 3\}$.
3) Apply mean removal to each block:

$$B'_{i,j} = B_{i,j} - \mu_{i,j},\ \mu_{i,j} = \frac{1}{256}\sum_{m=1}^{16}\sum_{n=1}^{16} B_{i,j}(m,n) \quad (3)$$

where$B'_{i,j}$ is the mean-removed block.

Step 2: CNN-Based Prediction

The preprocessed CTU, Y'_{CTU}, comprising mean-removed blocks, is fed into a CNN to predict splitting decisions at three depth levels ($l = 1,2,3$). The CNN architecture is defined as follows:

1) Convolutional Layers:

Layer 1: Apply 8 convolutional kernels of size 4 × 4, followed by ReLU activation:

$$F_1 = ReLU(W_1 * Y'_{CTU} + b_1), F_1 \epsilon \mathbb{R}^{16\times16\times8} \quad (4)$$

where $*$ denotes convolution, W_1 are the kernel weights, and b_1 is the bias.

Layer 2: Apply 16 convolutional kernels of size4 × 4, followed by ReLU activation:

$$F_2 = ReLU(W_2 * F_1 + b_2), F_2 \epsilon \mathbb{R}^{4\times4\times16} \quad (5)$$

Layer 3: Apply 32 convolutional kernels of size 2 × 2, followed by ReLU activation:

$$F_3 = ReLU(W_3 * F_2 + b_3), F_3 \epsilon \mathbb{R}^{2\times2\times32} \quad (6)$$

2) Feature Vector Construction:

Flatten the feature maps F_1, F_2, and F_3into a vector:

$$V = Flatten\ F_1 + Flatten\ F_2 + Flatten\ F_3, \quad V \epsilon \mathbb{R}^{2432} \quad (7)$$

3) Fully Connected Branches:

For each depth level $l \in \{1,2,3\}$, process V through a branch of three fully connected layers:

$$H_l^{(1)} = ReLU\left(W_l^{(1)}V + b_l^{(1)}\right) \quad (8)$$

$$H_l^{(2)} = ReLU\left(W_l^{(2)}H_l^{(1)} + b_l^{(2)}\right) \quad (9)$$

$$S_l = \sigma\left(W_l^{(3)}H_l^{(2)} + b_l^{(3)}\right) \quad (10)$$

where σ is the sigmoid activation function, and $S_l \epsilon [0,1]$ represents the probability of splitting at level l.

Incorporate the quantization parameter QP as an external feature by concatenating it to the input of the first fully connected layer in each branch:

$$\tilde{V} = [V, QP] \quad (11)$$

4) Binary Decision:

Threshold the sigmoid output:

$$D_l = \begin{cases} 1\ if\ S_l \geq 0.5\ (split), \\ 0 \quad otherwise\ (non\text{-}split). \end{cases} \quad (12)$$

5) Early Termination:

If $D_1 = 0$, skip predictions for levels 2 and 3, and set $D_2 = D_3 = 0$.

If $D_2 = 0$, skip prediction for level 3, and set $D_3 = 0$.

Step 3: Integration with HEVC

1) Replace the RDO process in the HEVC HM 16.12 reference software with the CNN-predicted splitting decisions $\{D_1, D_2, D_3\}$.

2) Encode the CTU using parameters specified in table II of the manuscript.

Output the predicted intra-coding mode and compute encoding time and RD performance metrics.

A. Dataset

To train our deep neural network for predicting the optimal intra-coding mode for Coding Tree Units (CTUs) in HEVC, we require a large dataset of CTU samples that cover a wide-range of content types and complexity levels. We make use of well-known datasets like RAISE [36], DIV2K [37], and UCID [38], which have been widely used in machine learning research. These datasets offer a diverse collection of images that enable the training of robust algorithms. Furthermore, advancements in hardware design have led to increased processing power, allowing machine learning approaches to excel in the field of image coding and video compression.

In our experiments, we used resolutions from the RAISE dataset (e.g., 4928×3264) and JCT-VC test set (e.g., 2560×1600 for Class A), representing high-quality raw images and standard video sequences, respectively. The RAISE dataset consists of a large-scale database containing 8,156 uncompressed images at various resolutions. It is available for download in four subsets, with 1,000, 2,000, 4,000, and 6,000 raw images, respectively. We select the subset with 2,000 images, each having a resolution of 4928×3264 pixels. In our research, we randomly divide a specific subset into three distinct sets, namely, training (85%), validation (5%), and test (10%), with CTUs extracted and labelled via HM 16.12. JCT-VC sequences were similarly processed as video frames. For the purpose of All-Intra (AI) encoding, we employ four quantization parameters (22, 27, 32, 37) in conjunction with the file encoder_ intra_main.cfg [39]. Our encoding approach involves assigning binary labels to each coding unit, which effectively indicates whether the unit is subjected to splitting (=1) or non-splitting (=0) during the process. Binary labels were assigned to each CU based on the ground-truth partitioning from HM 16.12's RDO process, serving as training targets for the CNN.

By adhering to these rigorous steps of partitioning and employing the specified quantization parameters, we ensure the robustness and efficacy of our proposed Fast CTU-Based Intra-coding method. Additionally, utilizing binary labels for the coding units facilitates a streamlined deep learning approach, enabling us to achieve efficient and accurate results for HEVC-based applications.

B. Proposed Model

The HEVC standard, introduced in 2013, has been a significant advancement in image and video coding technology. However, the complexity of the HEVC framework has resulted in increased encoding delays. One of the main contributors to this complexity is the intricate process of coding unit selection in intra-coding. While deep learning has shown promise in various domains, its potential for balancing compression efficiency and encoding complexity in HEVC remains largely unexplored. Convolutional neural networks (CNNs) have demonstrated effectiveness in feature extraction and classification tasks for videos and images. To address the challenges posed by the complexity of HEVC's intra mode, we propose a deep learning-based approach that leverages CNNs to predict coding unit (CU) partitions.

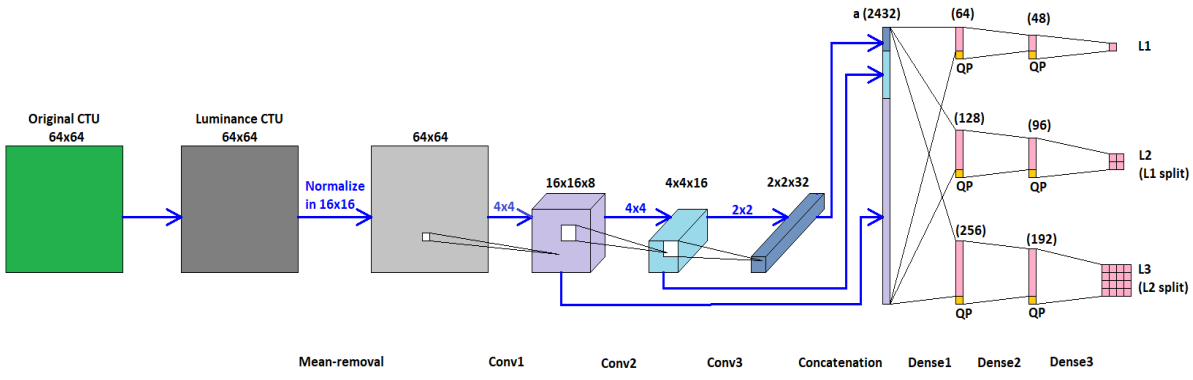

Fig. 3 Architecture of the proposed CNN model

Our proposed CNN model replaces the Rate-Distortion Optimization (RDO) technique used for CU selection in HEVC intra-coding. After training the model, we integrate it into the HEVC reference software. Figure 3 illustrates the structure of our proposed CNN model. The RAISE images converted to YUV format and encoded as independent frames, while JCT-VC sequences were processed as pseudo video sequences using HM 16.12.CTU samples were generated by processing raw YUV images from the RAISE dataset and JCT-VC sequences using HM 16.12, extracting 64×64 luminance blocks labelled with ground-truth splitting decisions from the original RDO process. Only the Y (Luminance) channel is utilized since it contains the majority of visual information. The proposed approach involves a series of interconnected layers. The CNN comprises three convolutional layers (8×4×4, 16×4×4, 32×2×2 kernels) for feature extraction, followed by three fully connected branches for multi-level splitting decisions, optimized for HEVC intra-coding's quadtree structure.

1) *Preprocessing layer:* The luminance CTU extracted from the original CTU is preprocessed through mean removal. We choose a 16×16 block for mean removal to align with the last splitting decision that can be taken at the 16×16 coding unit level. By subtracting the mean intensity value of each 16×16 block from its corresponding intensity values, we achieve a reduced variation in intensity within the input CTU samples.
2) *Convolutional layers:* The preprocessed input passes through three convolutional layers. To extract the low-level characteristics of CU partitions, we apply eight 4×4 kernels in the foremost convolutional layer. For the second convolutional layer, we employ sixteen 4×4 kernels to extract intermediate features. Finally, in the third convolutional layer, we further convolute the feature maps using 2×2 kernels to produce higher-level features. Each convolution layer's kernel size is explicitly denoted along the connecting lines that link these layers in Figure 3.Rectified linear units (ReLU) are applied as activation functions to all three convolutional layers.
3) *Concatenating layer:* The output feature maps from the initial three convolutional layers are concatenated to create a single vector that effectively captures both global and local characteristics. This vector encompasses a total of 2,432 concatenated features, essential for subsequent processing.
4) *Fully connected layers:* The concatenated features undergo processing through three branches of fully connected dense layers. At levels one, two, and three, respectively, these branches are responsible of producing splitting choices. There are three dense layers in each branch. The output layer performs the vital task of creating a partition map as the CNN's final output, whereas the first two hidden dense layers consecutively produce feature vectors. The labels in the partition map are binary to guarantee efficient learning. As a result, the output layer is activated using the sigmoid function, which works well for binary classification tasks, while the rectified linear units (ReLU) activation function is used for the two hidden dense layers.

We incorporate the quantization parameter (QP) as an external feature in the fully connected layers so that the CNN can adapt to various quantization values. The decision-making process at level one is one of the key aspect covered in our study. The execution of dense layers at levels two and three is effectively avoided when the choice at level one indicates a non-splitting scenario. Similarly, the execution of the dense layers at level three is stopped if all four outputs of level two result in a non-splitting decision. During the CNN process, these early termination methods reduce computational time.

TABLE II. ENCODER PARAMETER SETTINGS

Parameter	**Setting**
HEVC Reference Software	HM 16.12
Encoder Profile	Main
Configuration file	Encoder_intra_main.cfg
R/D Optimization	On
Motion Estimation	Hadamard
Search Range	64
Rate Control	Off
Coding unit(size/depth)	64/4
Transform unit size (min / max)	4/32
Quantization parameters	22, 27, 32, 37
Entropy coding method	CABAC
Frame/Field coding	Frame based coding

C. *Input Parameters*

In the context of employing the HEVC reference software for encoding input raw images, various essential encoder parameters demand precise specification. These critical parameters encompass the input video sequence,

video resolution, target bit rate, quantization step size (QP), and profile. By carefully defining these input parameters, the proposed approach can seamlessly integrate with the existing HEVC reference software and offer substantial enhancements in terms of compression efficiency and computational speed.

Table II provides details about the HEVC encoder settings used. We conducted an extensive QP analysis with values of 22, 27, 32, and 37. This range of QP values allows us to evaluate the performance of our proposed CTU-based intra-coding technique across a spectrum of compression levels, ensuring that our approach remains adaptable to various application scenarios and quality requirements. The fine-tuning of QP values is a critical aspect of achieving the desired balance between compression efficiency and visual fidelity in our deep learning-enhanced HEVC encoding framework.

V. Experimental Results and Analysis

We evaluated the performance of our proposed method by comparing it with the conventional HEVC intra-coding method under HEVC common test conditions. This section presents the experimental results and analysis of our approach, discussing the configuration, settings, and performance evaluation in relation to existing methods. To assess the effectiveness of our approach, we conduct a comparative analysis with two existing methods: the LeNet-5 approach introduced in [30] and the OSDN approach proposed in [34] for intra-coding of HEVC.

A. Configuration and Settings

To ensure a fair and accurate comparison, we implemented the LeNet, OSDN, and our proposed method within the HEVC reference software HM 16.12. For all evaluations, we utilized the default configuration file "encoder_intra_main.cfg" [39], which is well-suited for assessing the performance of intra-mode HEVC. Our experiments were conducted on a computer equipped with an Intel(R) Core (TM) i7-6500U CPU @ 2.60 GHz, 8 GB RAM, and Windows 10 Home 64-bit operating system.

During the evaluation process, we compressed raw images of various resolutions using four different Quantization Parameter (QP) values: 22, 27, 32, and 37.BD-BR (%) and BD-PSNR (dB) were computed per Bjøntegaard [40] to effectively assess the rate-distortion (RD) performance, where BD-BR measures average bit-rate increase and BD-PSNR quantifies PSNR change over a range of QPs.

Furthermore, we also introduced a measure of complexity reduction, denoted as ΔT, to quantify the encoding time reduction achieved by our proposed models. The calculation of ΔT is defined by Equation (13):

$$\Delta T = (T_{HM} - T_{CNN})/T_{HM} \times 100\% \quad (13)$$

Here, T_{HM} represents the encoding time of HM 16.12, and T_{CNN} represents the encoding time of our proposed models. By conducting a comprehensive evaluation with these configurations and settings, we seek to establish the superiority of our method over the LeNet-5 and OSDN approaches.

B. Performance Evaluation

During our experiments, models were trained with an initial learning rate of 0.01, using binary cross-entropy loss, Adam optimizer (step size 0.001), and 50 epochs, achieving a final loss of 0.15. Training took ~10 hours, with inference at ~5 ms per CTU. To facilitate the output decision process, we judiciously employed a threshold of 0.5. This particular threshold value was meticulously chosen, as it remarkably minimized any potential degradation in the RD (Rate-Distortion) performance. Our model achieved CU partition prediction accuracies of 90.89%, 87.24%, and 81.92% for decision levels one, two, and three, respectively, which are quite high. In this study, we have conducted experiments using various test images, and the results of encoding time saving ΔT are presented in table III. The reported ΔT values include CNN inference time, measured on an Intel i7-6500U, ensuring practical computational gains. We can observe from Fig. 4 that our proposed approach consistently outperforms both approaches LeNet-5 and OSDN in terms of encoding time savings for all tested resolutions and QP values.

For example, at QP = 22, our proposed approach achieves encoding time savings of 64.78% on average, while LeNet-5 and OSDN achieve 58.83% and 58.55%, respectively. Similarly, our approach outperforms the LeNet-5 and OSDN approaches at other QP values. Across all resolutions and QP values, our proposed approach achieves an average encoding time saving of 68.68%. In contrast, LeNet-5 achieves an average saving of 56.52%, and OSDN achieves 62.93%. This demonstrates the superiority of our proposed approach in terms of time efficiency. At each resolution, our approach consistently outperforms both approaches in terms of encoding time savings. For example, at the resolution of 4928×3264 and QP value 37, our proposed approach achieves 79.32% of time savings, while reference LeNet-5 and OSDN achieve 65.78% and 70.44%, respectively. We

achieved average time reductions of 64.78%, 67.65%, 69.90%, and 72.41% for QP values 22, 27, 32, and 37, respectively. Our proposed approach demonstrates remarkable time efficiency improvements across all tested resolutions and QP values.

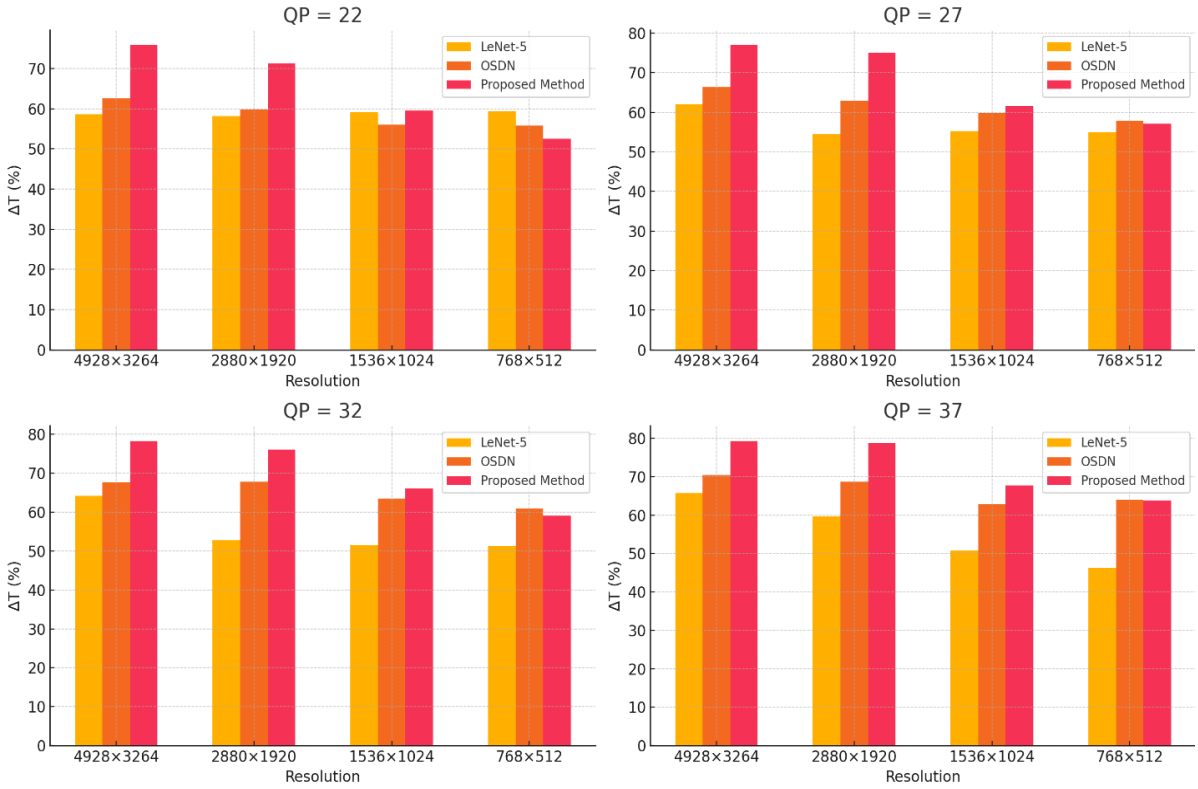

Fig. 4 Encoding time savings (ΔT) for different resolutions and QP values

It is crucial to highlight the importance of these time savings as it can lead to more efficient video compression and improved real-time processing capabilities in various applications. Our model outperformed the other two approaches in terms of encoding time reduction for intra-mode and fast splitting decision. Notably, our model exhibited remarkable performance for high-resolution images. Furthermore, as the QP increased, our approach consistently outperformed the other two methods. This can be attributed to the fact that as the QP increases, larger coding units (CUs) are used, and our approach streamlines the decision-making process by bypassing level two and level three splitting when level one remains non-split, thereby enhancing the overall coding efficiency. By avoiding unnecessary splitting decisions, our method aims to significantly reduce computational complexity while maintaining or even improving coding performance. Consequently, employing our CNN model in the HEVC reference software requires less encoding time.

In conclusion, the results in table III indicate that our proposed CNN model for fast intra-coding in HEVC outperforms existing methods (LeNet-5 and OSDN) in terms of encoding time savings across various resolutions and QP values. The average time savings achieved by our approach are substantial, demonstrating the effectiveness of using deep learning for improving the intra-coding process in HEVC.

TABLE III. ENCODING TIME SAVING COMPARISON

Resolution	Encoding time saving ΔT (%)											
	QP = 22			QP = 27			QP = 32			QP = 37		
	LeNet-5	OSDN	Our	LeNet-5	OSDN	Our	LeNet-5	OSDN	Our	LeNet-5	OSDN	Our
4928×3264	58.66	62.53	**75.83**	62.05	66.33	**76.99**	64.23	67.66	**78.33**	65.78	70.44	**79.32**
2880×1920	58.12	59.84	**71.25**	54.42	62.90	**75.03**	52.74	67.86	**76.05**	59.67	68.67	**78.79**
1536×1024	59.13	55.97	**59.49**	55.30	59.85	**61.52**	51.55	63.49	**66.12**	50.72	62.85	**67.72**
768×512	**59.39**	55.84	52.54	54.95	**57.89**	57.06	51.31	**60.92**	59.09	46.28	**63.90**	63.82
Average	58.83	58.55	**64.78**	56.68	61.74	**67.65**	54.96	64.98	**69.90**	55.61	66.47	**72.41**

TABLE IV. RD PERFORMANCE COMPARISON

Resolution	BD-BR (%)			BD-PSNR (dB)		
	LeNet-5	OSDN	Our	LeNet-5	OSDN	Our
4928×3264	3.956	1.805	**1.271**	-0.300	-0.132	**-0.081**
2880×1920	4.182	2.765	**1.490**	-0.294	-0.141	**-0.088**
1536×1024	5.564	3.212	**1.205**	-0.348	-0.265	**-0.092**
768×512	5.255	3.105	**0.985**	-0.335	-0.218	**-0.075**
Average	4.739	2.722	**1.238**	-0.319	-0.189	**-0.084**

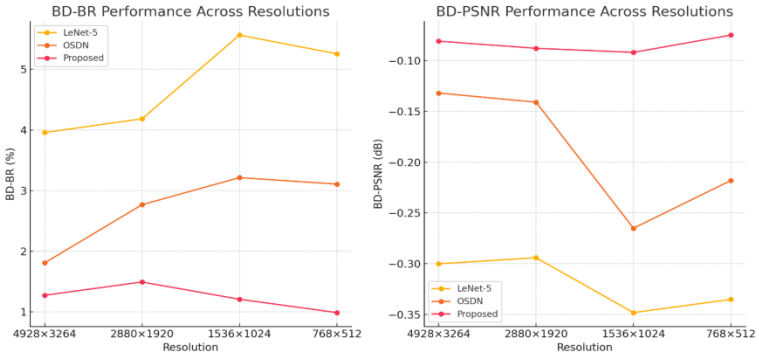

Fig. 5 RD performance comparison across various resolutions

The table IV illustrates the RD performance of our approach compared to the other two methods. On average, our CNN model exhibited a 1.238% increase in BD-BR compared to 4.739% and 2.722% in LeNet-5 and OSDN, respectively. Our implementation of LeNet-5 yielded a higher BD-BR (4.739%) than reported in [30] (1.1%), despite similar time savings. This discrepancy may stem from differences in training datasets (RAISE vs. their custom set) and HM configurations, highlighting the sensitivity of CNN performance to data and implementation details. This indicates that our method can achieve better compression efficiency, leading to smaller file sizes for the same visual quality. Our proposed approach achieved an average BD-PSNR of -0.084 dB, outperforming existing methods such as LeNet-5, which scored -0.319 dB, and OSDN, which achieved -0.189dB.This suggests that our method produces compressed images with higher visual quality compared to the reference methods. In summary, our proposed approach shows better results in terms of both compression efficiency (BD-BR) and visual quality (BD-PSNR) when compared to the existing methods (LeNet-5 and OSDN). This indicates that our CNN model is effective in fast intra coding for HEVC and provides promising results for video compression applications. Fig. 5 effectively highlights the superior performance of our proposed approach in terms of both compression efficiency and visual quality.

TABLE V. ENCODING TIME SAVING COMPARISON FOR THE JCT-VC TEST SET (AI)

Class	Video	Encoding time saving ΔT (%)											
		QP = 22			QP = 27			QP = 32			QP = 37		
		LeNet-5	OSDN	Our	LeNet-5	OSDN	Our	LeNet-5	OSDN	Our	LeNet-5	OSDN	Our
A	PeopleOnStreet	52.05	51.89	**58.95**	51.65	53.88	**58.88**	38.36	56.05	**62.31**	35.69	61.08	**64.28**
	Traffic	39.45	54.67	**65.95**	27.45	58.56	**69.93**	21.62	63.57	**72.13**	33.28	67.84	**74.26**
B	BasketballDrive	47.84	68.22	**73.64**	39.05	67.31	**75.84**	41.29	68.37	**77.42**	48.54	69.61	**78.34**
	BQTerrace	**66.56**	51.25	51.96	62.35	57.77	**66.30**	60.38	60.84	**67.46**	38.57	61.64	**72.64**
	Cactus	39.64	**57.53**	49.30	42.85	**59.67**	54.45	45.28	64.51	**67.84**	61.05	68.56	**73.89**
	Kimono	34.58	70.39	**79.15**	45.28	70.85	**83.51**	51.46	73.84	**83.92**	64.55	73.85	**83.67**
	ParkScene	43.35	60.58	**61.10**	44.58	65.38	**66.95**	61.54	68.47	**69.86**	64.51	70.64	**74.60**
C	BasketballDrill	46.90	**58.88**	43.31	58.87	**60.57**	46.85	48.56	**64.08**	63.29	62.34	**65.77**	63.25
	BQMall	52.36	47.55	**53.32**	44.01	49.85	**56.85**	36.58	53.65	**60.50**	55.48	57.95	**65.53**
	PartyScene	**63.54**	54.06	38.79	49.80	**55.90**	42.63	32.95	**59.45**	44.55	28.54	**64.59**	56.08
	RaceHorses	44.72	50.24	**53.93**	42.52	**57.73**	55.65	43.55	**59.99**	59.62	50.07	**63.01**	62.07
D	BasketballPass	45.57	**58.91**	55.31	41.57	**60.39**	57.43	39.68	**62.59**	57.08	38.67	**64.56**	58.64
	BlowingBubbles	**55.68**	54.65	40.53	42.55	**58.95**	40.09	27.58	**60.35**	40.89	26.54	**63.54**	46.98
	BQSquare	**59.87**	44.06	43.90	**60.35**	44.58	46.08	**58.94**	48.87	47.06	46.85	**49.68**	48.84
	RaceHorses	43.66	**52.64**	51.89	40.57	**55.98**	55.15	41.54	**58.85**	57.95	40.54	60.08	**60.34**
E	FourPeople	53.64	56.52	**64.81**	42.54	61.84	**71.87**	28.54	64.62	**73.58**	26.34	67.31	**75.09**
	Jonny	58.29	62.35	**69.56**	60.34	65.09	**71.45**	63.92	67.62	**71.94**	68.68	70.65	**72.43**
	KristenAndSara	54.65	61.38	**73.08**	58.24	64.55	**73.99**	56.34	65.35	**76.23**	62.58	65.95	**75.83**
Average		50.13	56.43	**57.14**	47.48	59.38	**60.77**	44.34	62.28	**64.09**	47.38	64.8	**67.04**

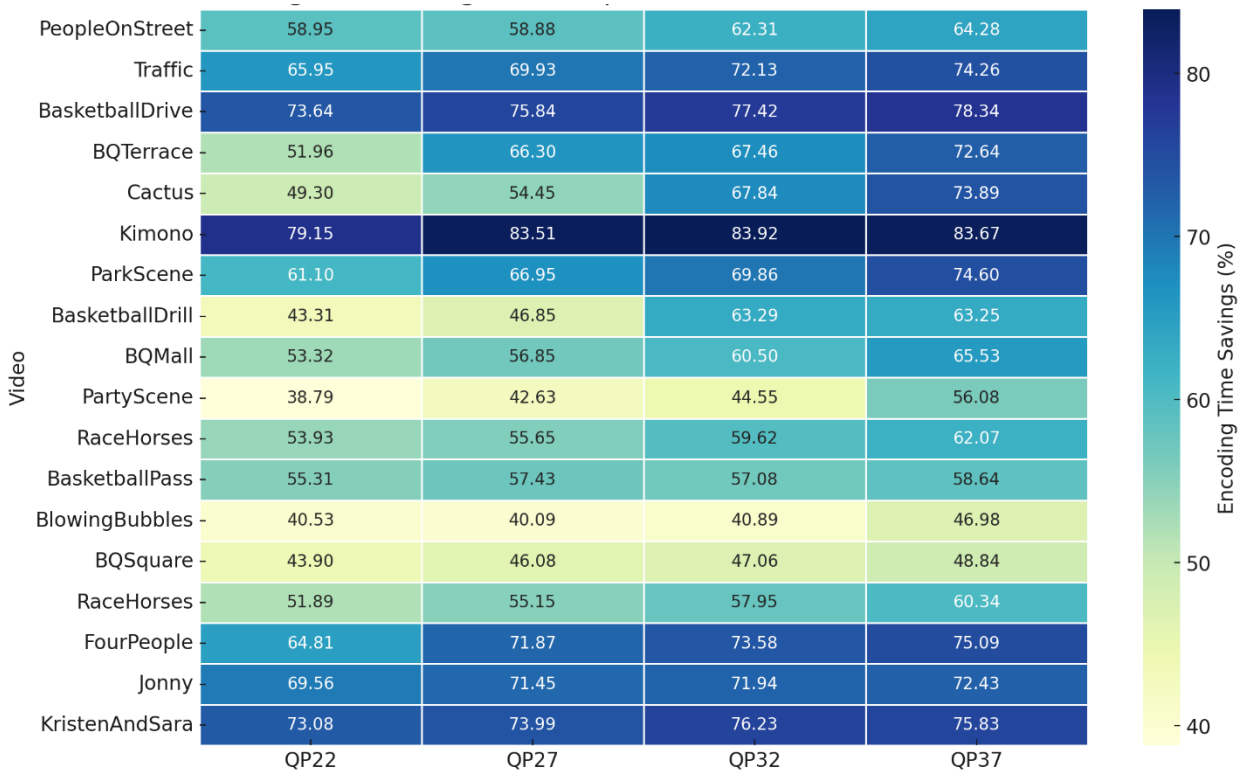

Fig. 6 Encoding time savings heatmap for the JCT-VC test set (AI)

The results are presented for three different approaches: LeNet-5, OSDN, and our proposed method. For the "PeopleOnStreet" video, our proposed method achieved a significant encoding time saving of 58.95%, 58.88%, 62.31%, and 64.28% at QP values of 22, 27, 32, and 37, respectively, outperforming both LeNet-5 and OSDN. In the case of the "Traffic" video, our proposed method consistently outperformed LeNet-5 and OSDN, achieving encoding time savings of 65.95%, 69.93%, 72.13%, and 74.26% across different QP values.

Results for Classes B, C, D, and E follow a similar pattern, with our proposed method consistently demonstrating superior encoding time savings compared to the baselines across various videos and QP values. The average encoding time saving across all classes and videos for our proposed method at QP value of 37 was 67.04%, outperforming both LeNet-5 (47.38%) and OSDN (64.8%). The heatmap in Fig. 6 provides a clear visual representation of how our proposed method performs in terms of encoding time savings for each video sequence at various QP levels. The color intensity reflects the percentage of time saved, with darker shades indicating higher savings.

Comparing the data presented in tables III and V, it becomes evident that the disparity in time savings between our method and alternative approaches widens as the QP value rises. This phenomenon is presumably attributed to the escalation in QP, which correlates with an increase in larger Coding Units (CUs). Consequently, our CNN model's early termination mechanism can effectively operate on a greater number of Coding Tree Units (CTUs) due to the prevalence of larger CUs, thereby reducing overall processing time.

Table VI summarizes the rate-distortion (RD) performance comparison for the JCT-VC test set. From the results, it is evident that our proposed approach outperforms both LeNet-5 and OSDN in terms of encoding time savings and minimal increase in bit rate while maintaining lower distortion levels across various video classes. The two bar charts in Fig. 7 illustrate the differences in compression efficiency and video quality between the approaches. The lower the BD-BR and the closer the BD-PSNR is to zero, the better the performance in terms of rate-distortion. The top chart shows the BD-BR comparison, highlighting how our method consistently achieves lower bit rate increases. The bottom chart compares BD-PSNR, where our method also performs better by maintaining lower distortion levels across the video classes.

On average, our approach achieved a BD-BR of 2.361% and a BD-PSNR of -0.107dB.Hence, our method stands out as the top performer among the three strategies concerning Rate-Distortion (RD) performance. The significant enhancement in RD observed in our method primarily stems from the exceptional accuracy in predicting Coding Unit (CU) partitioning. This proficiency is achieved through a deep CNN architecture, which has been trained extensively on a vast dataset, allowing it to learn an ample number of parameters effectively.

TABLE VI. RD PERFORMANCE COMPARISON FOR THE JCT-VC TEST SET (AI)

Class	Video	BD-BR (%)			BD-PSNR (dB)		
		LeNet-5	OSDN	Our	LeNet-5	OSDN	Our
A	PeopleOnStreet	9.543	4.856	**2.209**	-0.495	-0.247	**-0.132**
	Traffic	5.756	4.982	**2.653**	-0.328	-0.286	**-0.130**
B	BasketballDrive	8.955	6.047	**4.351**	-0.250	-0.145	**-0.119**
	BQTerrace	6.871	4.982	**2.088**	-0.298	-0.281	**-0.092**
	Cactus	6.982	6.024	**2.326**	-0.261	-0.201	**-0.083**
	Kimono	5.134	**2.505**	2.607	-0.175	**-0.089**	-0.091
	ParkScene	3.666	3.457	**2.470**	-0.169	-0.138	**-0.079**
C	BasketballDrill	9.895	10.983	**3.211**	-0.444	-0.567	**-0.128**
	BQMall	9.551	8.005	**1.957**	-0.509	-0.482	**-0.107**
	PartyScene	7.005	7.982	**1.005**	-0.456	-0.704	**-0.050**
	RaceHorses	6.760	4.652	**2.106**	-0.390	-0.265	**-0.112**
D	BasketballPass	9.586	8.004	**1.957**	-0.546	-0.485	**-0.113**
	BlowingBubbles	6.009	8.452	**0.782**	-0.372	-0.485	**-0.041**
	BQSquare	10.985	2.755	**1.048**	-0.866	-0.300	**-0.070**
	RaceHorses	8.685	5.559	**1.459**	-0.501	-0.352	**-0.084**
E	FourPeople	9.005	8.051	**2.942**	-0.488	-0.456	**-0.167**
	Jonny	10.856	7.982	**3.954**	-0.462	-0.354	**-0.147**
	KristenAndSara	11.954	5.628	**3.367**	-0.598	-0.259	**-0.174**
Average		8.178	6.161	**2.361**	-0.423	-0.339	**-0.107**

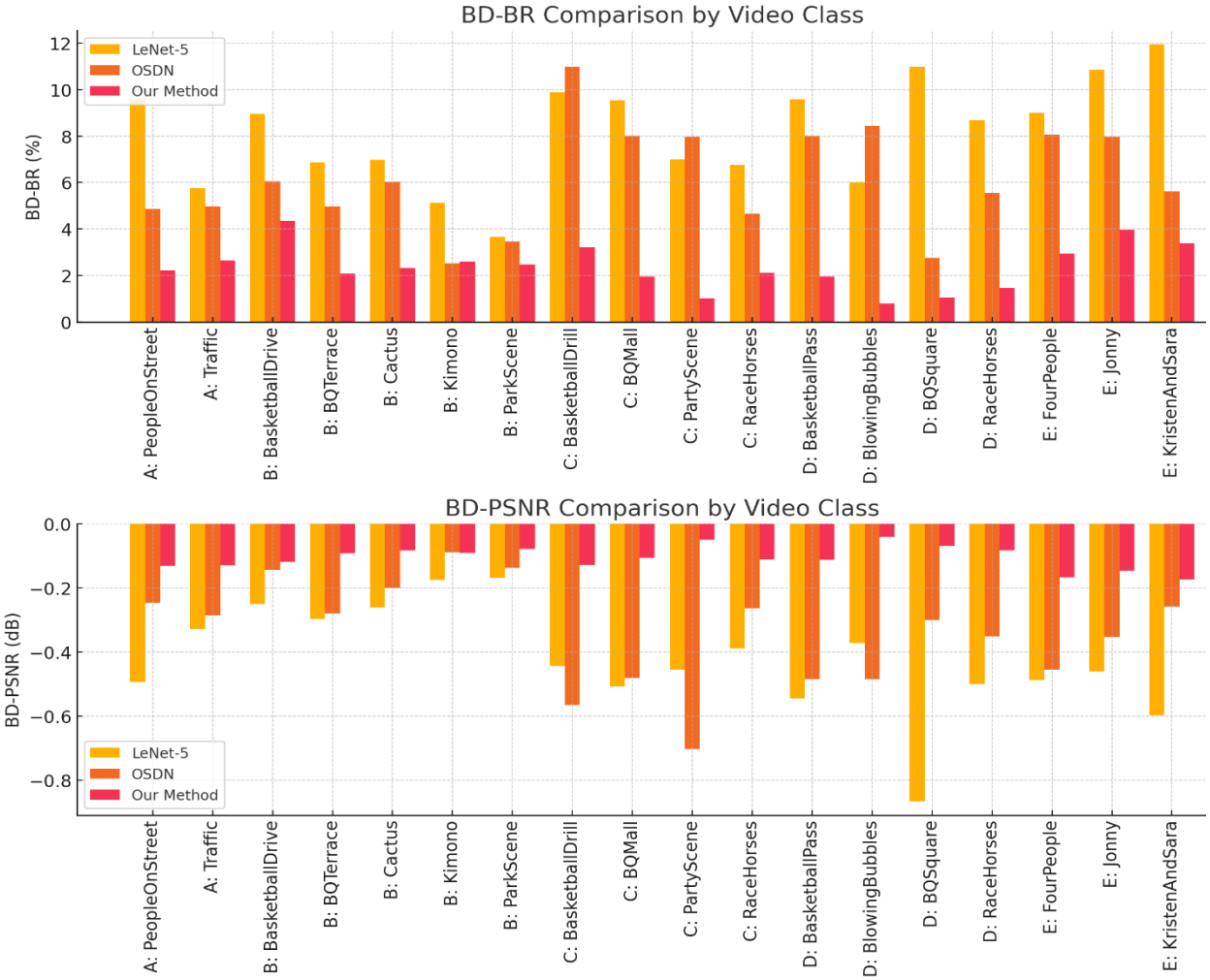

Fig. 7 RD performance comparison by video class for the JCT-VC test set (AI)

By leveraging neural networks, our method efficiently identifies and encodes the intra-prediction modes, leading to significant time savings without compromising the quality of encoded videos. Moreover, the adaptability of our approach across different video classes showcases its robustness and generalizability.

Furthermore, the distortion results indicate that our approach successfully preserves the visual quality of encoded videos, making it suitable for various applications where maintaining high perceptual quality is crucial.

By leveraging a deep CNN structure to learn how to anticipate the CTU split, we effectively decrease the intra-mode complexity of HEVC. Our proposed method strategically terminates the CTU partition early, leveraging insights from partitioning structures to minimize computational costs. The experimental results reveal that our CNN model is capable of making rapid intra-mode decisions and outperforms other existing approaches. Specifically, our proposed model achieves a remarkable reduction in encoding time, approximately 69%, with negligible loss in rate-distortion (RD) performance. The ability to accelerate the CTU-based intra-coding process in HEVC using deep learning has significant implications for real-time video applications on resource-constrained devices. In conclusion, our CNN model outperformed the other two models in terms of intra-mode encoding time reduction and RD performance. Moreover, objective quality assessment computed as BD-PSNR, demonstrate that our approach produces visually pleasing reconstructed videos.

VI. CONCLUSION

In this manuscript, we have presented a novel approach for fast CTU-based intra-coding in HEVC using a deep learning methodology. New partitioning algorithms (QTMTT) are introduced with the transition to Versatile Video Coding (VVC). Although our approach focuses on HEVC, its CNN-based prediction may be able to adjust to the complexity of VVC, providing a basis for further cross-standard optimization. Our experimental results have shown that our suggested approach maintains competitive coding efficiency while achieving a notable reduction in computational complexity. We are convinced that our method has enormous potential for providing real-time HEVC video compression on a variety of devices, which will enhance video streaming and communication applications.

To sum up, our study shows how well a deep learning method works for quick CTU-based intra-coding in HEVC. Our approach is positioned as a promising solution for real-time video compression in a variety of devices due to the significant reduction in computing complexity and competitive coding efficiency. Our goal is to improve HEVC's overall video compression capabilities while also advancing video streaming and communication applications. Reliance on HM 16.12 and possible over fitting to RAISE/JCT-VC data are among the limitations. Future work could explore VVC adaptation, larger datasets, and hardware acceleration.

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.

Funding

The authors declare that no funding has been disclosed for this research.

REFERENCES

[1] Shiguo Lian, Zhongxuan Liu, Zhen Ren, and Haila Wang, “Commutative Encryption and Watermarking in Video Compression,” IEEE Transaction on Circuits and Systems for Video Technology, vol. 17, no. 6, pp. 774-778, June 2007.

[2] Cisco, Inc., “Cisco Annual Internet Report,” 2018–2023.

[3] B. R. Maheshwari, “Exploring the Potential of Deep Learning in Overcoming Challenges Faced by Rural Kutch,” International Journal of Novel Research and Development, vol. 8, issue5, May 2023.

[4] S. Momcilovic, N. Roma, L. Sousa, and I. Milentijevic, “Run-time Machine Learning for HEVC/H.265 Fast Partitioning Decision,” 2015 IEEE International Symposium on Multimedia, pp. 347-350, 2015.

[5] Z. Liu, X. Yu, Y. Gao, S. Chen, X. Ji, and D. Wang, “CU partition mode decision for HEVC hardwired intra encoder using convolution neural network,” IEEE Transactions on Image Processing, vol. 25, no. 11, pp. 5088–5103, 2016.

[6] Y. Zhang, S. Kwong, X. Wang, H. Yuan, Z. Pan and L. Xu, “Machine learning-based coding unit depth decisions for flexible complexity allocation in high efficiency video coding,” IEEE Transactions on Image Processing, vol. 24, no. 7, pp. 2225–2238, Jul. 2015.

[7] Z. Jin, P. An, L. Shen, and C. Yang, “CNN oriented fast QTBT partition algorithm for JVET intra coding,” in VCIP. IEEE, 2017, pp. 1–4.

[8] J. Xu, M. Xu, Y. Wei, Z. Wang, and Z. Guan, “Fast H.264 to HEVC transcoding: A deep learning method,” IEEE Transactions on Multimedia, DOI: 10.1109/TMM.2018.2885921, 2018.

[9] N. Song, Z. Liu, X. Ji, and D. Wang, “CNN oriented fast PU mode decision for HEVC hardwired intra encoder,” in GlobalSIP. IEEE, 2017, pp. 239–243.

[10] Q. Hu, Z. Shi, X. Zhang and Z. Gao, "Fast HEVC intra mode decision based on logistic regression classification," in 2016 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), 2016.

[11] D. Liu, X. Liu and Y. Li, "Fast CU size decisions for HEVC intra frame coding based on support vector machines," in 2016 IEEE 14th Intl Conf on Dependable, Autonomic and Secure Computing (DASC), 2016.

[12] X. Shen and L. Yu, "CU splitting early termination based on weighted SVM," EURASIP J Image Video Process, vol. 2013, no. 1, 2013.

[13] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," in The Handbook of Brain Theory and Neural Networks, M. Arbib, Ed. Cambridge, MA, USA: MIT Press, 1995, pp. 255–258.

[14] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proc. IEEE, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[15] D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber, "Convolutional neural network committees for handwritten character classification," in Proc. 11th IEEE Int. Conf. Document Anal. Recognit. (ICDAR), Sep. 2011, pp. 1135–1139.

[16] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in Proc. 25th IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2012, pp. 3642–3649.

[17] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2015, pp. 3431–3440.

[18] G. Tian and S. Goto, "Content adaptive prediction unit size decision algorithm for HEVC intra coding," in Picture Coding Symposium (PCS), Krakow, Poland, May. 2012, pp. 405–408.

[19] Y. Liu, X. Liu, and P. Wang, "A Texture Complexity Based Fast Prediction Unit Size Selection Algorithm for HEVC Intra-coding," IEEE 17th International Conference on Computational Science and Engineering (CSE), 2014, pp. 1585–1588.

[20] Z. Pan, S. Kwong, Y. Zhang, J. Lei, and H. Yuan, "Fast Coding Tree Unit depth decision for high efficiency video coding," IEEE International Conference on Image Processing (ICIP), 2014, pp. 3214– 3218.

[21] Y. Li, Y. Liu, H. Yang, and D. Yang, "Fast CU splitting and pruning method based on online learning for intra coding in HEVC," IEEE Visual Communications and Image Processing Conference, 2014, pp. 450–453.

[22] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," IEEE Transactions on Circuits and Systems for Video Technology, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[23] J. Ohm, Gary J. Sullivan, Thiow K. Tan, and T. Wiegand, "Comparison of the Coding Efficiency of Video Coding Standards-Including High Efficiency Video Coding (HEVC)," IEEE Transactions on Circuits and Systems for Video Technology, vol. 22, no. 12, pp. 1669–1684, Dec. 2012.

[24] L. M. Qin , Z. J. Zhu, Y. Q. Bai, G. L. Liao, and T. N. Liu, "A Complexity-Reducing HEVC Intra-Mode Method Based on VGGNet," Journal of Computers, vol. 33, no. 4, pp. 57-67, Aug. 2022.

[25] T. Li, M. Xu, and X. Deng, "A Deep Convolutional Neural Network Approachfor Complexity Reductionon Intra-mode HEVC," Proc. of the IEEE International Conference on Multimedia and Expo, pp. 1255-1260, Jul. 2017.

[26] S. Bouaafia, R. Khemiri, A. Maraoui, and F. E. Sayadi,"CNN-LSTM Learning Approach-Based Complexity Reduction for High-Efficiency Video Coding Standard," Scientific Programming, Vol. 2021, Article ID 6628041, pp. 1-10,2021.

[27] Linwei Zhu, Sam Kwong, Yun Zhang, Shiqi Wang, and Xu Wang, "Generative Adversarial Network Based Intra Prediction for Video Coding," IEEE Transaction on Multimedia, pp. 1-14, 2019.

[28] Z. Feng, P. Liu, K. Jia, K. Duan, "Fast Intra CTU Depth Decision for HEVC," IEEE Access 6,pp. 45262-45269, 2018.

[29] M. Xu, T.Y. Li, Z.L. Wang, X. Deng, R. Yang, Z. Guan, "Reducing complexity of HEVC: a deep learning approach," IEEE Transactions on Image Processing, vol. 27, no. 20, pp. 5044-5059, 2018.

[30] H.-C Ting, H.-L Fang, J.-S Wang, "Complexity Reduction on HEVC Intra Mode Decision with modified LeNet-5," in: Proc. of IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS), 2019.

[31] K. Kim, W.W. Ro, "Fast CU Depth Decision for HEVC Using Neural Networks," IEEE Transactions on Circuits and Systems for Video Technology, vol. 29, no. 5, pp. 1462-1473, 2019.

[32] Zhibo Chen , Jun Shi, and Weiping Li, "Learned Fast HEVC Intra Coding," IEEE Transactions on Image Processing, vol. 29, pp. 5431-5446, 2020.

[33] Farid Zaki, Amr E. Mohamed, and Samir G. Sayed, "CtuNet: A Deep Learning-based Framework for Fast CTU Partitioning of H265/HEVC Intra- coding," Ain Shams Engineering Journal, vol. 12 (2), pp. 1859-1866, 2021.

[34] Z.X. Wang, F. Li, "Convolutional neural network based low complexity HEVC intra encoder," Multimedia Tools and Applications, vol. 80, no. 2, pp. 2441-2460, 2021.

[35] Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11. (2016, Jun.). HM 16.12 Reference Software [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-16.12/

[36] Dang-Nguyen DT, Pasquini C, Conotter V, Boato G. “RAISE: a raw images dataset for digital image forensics,” in Proc. 6th ACM Multimedia Systems Conference on - MMSys 15, ACM Press; 2015. doi:10.1145/ 2713168.2713194.

[37] Timofte R, Agustsson E. NTIRE, “2017 challenge on single image super-resolution: Methods and results,” in 2017 IEEE conference on computer vision and pattern recognition workshops (CVPRW), IEEE; 2017. doi:10.1109/ cvprw.2017.149.

[38] Schaefer G, Stich M, “UCID: an uncompressed color image database,” in Yeung MM, Lienhart RW, Li CS, editors, Storage and retrieval methods and applications for multimedia 2004, SPIE; 2003. doi:10.1117/12.525375.

[39] F. Bossen, “Common test conditions and software reference configurations,” Joint Collaborative Team on Video Coding, document Rec. JCTVC-L1100, Jan. 2013.

[40] G. Bjøntegaard, “Calculation of avarage PSNR difference between RD-curves,” in ITU-T, VCEG-M33, Austin, TX, USA, Apr. 2001.

CHENA RAM received his Bachelor of Engineering degree in Electronics and Communication Engineering from the University of Rajasthan, Jaipur, India, in 2004, and his Master of Engineering degree in Computer Science and Engineering from Punjab University, Chandigarh, India, in 2014. He is currently pursuing a PhD degree in the Department of Computer Engineering at Rajasthan Technical University, Kota, India. Since 2004, he has been actively engaged in research and teaching in the Department of Electronics and Communication Engineering at Government Engineering College, Bikaner, India. His research interests span image and video processing, deep learning, computer networks, and information theory and coding.

SUBHASH PANWAR obtained his Bachelor of Engineering degree in Computer Science and Engineering from the University of Rajasthan, Jaipur, India, in 2004, followed by a Master of Technology degree in Computer Engineering from Motilal Nehru National Institute of Technology, Allahabad, India, in 2010, and a PhD in Computer Engineering from the Malaviya National Institute of Technology, Jaipur, India, in 2015. Since September 2004, he has been a faculty member in the Department of Information Technology at Government Engineering College, Bikaner, Rajasthan, India. His research focuses on areas such as image fusion, computer vision, computational intelligence, machine learning, and pattern recognition.