Received: October 19, 2024. Revised: December 5, 2024.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Coding Unit Partitioning Using Refined Stacked – Convolutional Neural Network with Scaled Exponential Linear Unit

Revathi Kasinathaperumal[1]* **Hosanna Princye Periapandi[2]**

[1]*Department of Electronics and Communication Engineering, S.E.A College of Engineering and Technology, Visvesvaraya Technological University, Belagavi, India*
[2] *Department of Electronics and Communication Engineering, Sri Sairam College of Engineering, Visvesvaraya Technological University, Belagavi, India*
* Corresponding author's Email: revathiselvaraj1229@gmail.com

Abstract: High-Efficiency Video Coding (HEVC) is a significantly used video coding standard with primary encoding challenges in the searching process. Video encoding is a challenging task due to high complexity and time consumption in Coding Tree Unit (CTU) partitioning. The Refined Stacked – Convolutional Neural Network with Scaled Exponential Linear Unit (RS-CNN with SELU) method is developed to minimize complexity and time consumption of encoding process. Initially, the blocks are partitioned by using Quad Tree (QT) based portioning method. The partition of the Coding Unit (CU) is performed by using the RS-CNN with SELU method, which minimizes the overall complexity and quickens the process of encoding. By using the refined stacking layers in the CNN method, it learns complete feature representation which is essential for correct partitioning of CU. The SELU activation function is carried out in the training process of RS-CNN that maintains stable activations and leads to quick convergence for accurate CU partition. The RS-CNN with SELU method attains Bjontegaard Delta Bit Rate (BD-BR) of 2.569% on JCT-VC dataset and 3.320% of BD-BR on UVG dataset which shows effective performance than Densely Connected Convolutional Neural Network (D-CNN).

Keywords: Coding tree unit, High-efficiency video coding, Quad tree, Refined stacked – convolutional neural network, Scaled exponential linear unit.

1. Introduction

In recent years, the standards of video coding like Advanced Video Coding (AVC), High-Efficiency Video Coding (HEVC) and Versatile Video Coding (VVC) accomplish effective development [1-3]. The evolution from AVC to HEVC and VVC describes the significant advancements in video compression techniques. Each standard demonstrates effective improvements in compression efficiency, flexibility for high resolutions and support for advanced applications. All these standards employ block coding and introduce various effective tools for coding, which increase the complexity of video coding [4]. By considering the VVC and HEVC, the HEVC adopted quad-tree based Coding Unit (CU) partition technique for CU [5]. The VVC introduces a high computational complexity when compared to HEVC because of its advanced and effective encoding tools. In HEVC, a quad-tree structure is utilized to partition the CTUs into CUs, providing a flexible and simple partitioning technique. However, the VVC utilizes quad-tree, binomial and trinomial tree partition architectures for optimal CU, that makes complexity of intra-frame coding 18 times greater than the HEVC [6]. If Rate Distortion (RD) optimization is used, it further increases the complexity [7]. Hence, high coding complexity restricts VVC standards application in real-time scenarios. So, the HEVC is a majorly utilized, standard method in industrial applications than the VVC [8]. These standards adopt block-based encoding architectures with variance in particular technologies [9, 10].

The much flexible procedure of quad-tree partition is utilized in HEVC for enhancing coding effectiveness of intra and inter-frame prediction, including CU, Transform Units (TU) and Prediction Units (PU) [11-13]. Every image frame is separated into Coding Tree Units (CTUs) in HEVC, and CTU is separated to multi-CUs by quad-tree structure [14]. Though, with the development of UltraHigh-Definition video services and quick development of video utilization, extension of video information surpasses enhancement of compression ratio [15]. The UltraHigh-Definition handles the traditional method's limitations of high data volume, management of computational complexity, and quality preservation. Hence, this approach is effective for exploring the video compression methods which provide high-quality video with constraints of restricted network bandwidth and capacity of storage [16]. The remarkable achievements of Deep Learning (DL) based algorithms in the process of CU partitions show a promising future for further exploration in this field [17]. Video encoding is a challenging task because the process of CTU partitioning maximizes the complexity of the encoding process and consumes more time. The essential contributions of the research are given below:

- The Refined Stacked – Convolutional Neural Network with Scaled Exponential Linear Unit (RS-CNN with SELU) method for CU partitioning minimises the complexity and time in the encoding process by correctly predicting the optimal CU sizes and splitting the patterns.
- The refined stacked layers are used in CNN to learn the representation of features completely for accurate CU partition.
- The SELU activation function is utilized during the training process of RS-CNN which maintains stable activations, alongside fast convergence for accurate CU partition.

The research paper is organized as follows: Section 2 analyses literature review of existing research, Section 3 provides details of proposed technique, while Section 4 gives results and discussion of proposed methodology, and conclusion of this research paper is given in Section 5.

2. Literature review

In this section, the existing researches on HEVC in recent years is briefly discussed, along with its advantages and disadvantages.

Galiano [18] suggested the integration of two methods deep learning and parallel computing to minimize the complexity of HEVC encoding. The first acceleration method was a parallel scheme that utilized the domain decomposition method, based on slice partitioning of HEVC. This was specifically suited to exploit the shared memory parallelism of multiple-core processors. The suggested method utilized optimization techniques at the CTU level to minimize the complexity of quad-tree splitting. The suggested method minimized the complexity of CTU partitioning with parallel schemes. However, it consumed more time for the encoding process due to a larger number of frames.

Wang [19] presented the Densely Connected Convolutional Neural Network (D-CNN) for predicting CU partitions. Initially, densely connected block was developed to enhance accuracy of CU partitioning through completely extracting CTU pixel features. Next, Efficient Channel Attention (ECA) and adaptive convolution kernel size were employed for quick partition of CU, which captured the data of D-CNN convolution channels. At last, the strategy of threshold optimization was employed for choosing the optimal threshold for every depth to further balance execution complexity of video coding. The presented D-CNN method did not learn complete features of the CU partition due to a fewer number of layers.

Jin [20] introduced the Temporal Context-based Video Compression Network (TCVC-Net) to enhance performance of video compression. The module of Global Temporal Reference Aggregation (GTRA) was developed for attaining the correct temporal reference to predict motion-compensated through aggregating the context of long-term temporal. For compressing motion vectors and residue, the Temporal Conditional Codec (TCC) was developed to prevent the structural and data through exploiting the components of multiple frequencies in a temporal context. The introduced method had lesser prediction accuracy due to a fewer reference frames.

Guo [21] carried out enhanced context mining and filtering to improve compression efficiency of Deep Contextual Video Compression (DCVC) method. Initially, the DCVC context was produced without redundancy and supervision present between context channels. This method overcame the redundancy across context channels for attaining effective context features. Next, transformer-based enhancement network was introduced as filtering module to capture long-range dependencies by enhancing the compressing process. The method improved the compression efficiency, but it required a high amount of time for the encoding process due to multiple passes over the data for feature refining.

Wang [22] developed the Joint Random Forest Classifier (JRFC) for deciding on CU partition kinds and developed a quick adaptive CU partition decision algorithm for intra-coding. The developed method made partition decisions for CUs of different sizes and fully bypassed intricate process of Rate Distortion Optimization (RDO). Hence, this process effectively minimized the encoding time. The process of RDO in CTU partitioning required all possible combinations of partitions, which resulted in high complexity.

From the above analysis, the existing algorithms for CU partitioning are seen to exhibit high complexity, consume more time for the encoding process and lower the prediction accuracy. These limitations degrade the performance of CU partition in HEVC. To mitigate these limitations, this research proposes DL based algorithm that is RS-CNN with the SELU method for CU partition. The RS-CNN with SELU method effectively minimizes the complexity and time consumption, leading to high prediction performance of CU partition.

3. Proposed methodology

In this research, the RS-CNN with the SELU method minimizes the complexity and time consumption of the encoding process.

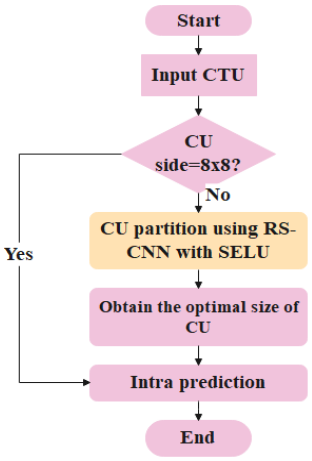

Figure. 1 Process of CU partition using RS-CNN with SELU

By stacking multiple conventional layers, the RS-CNNs develop the hierarchical representation of input data that simplifies the process of encoding in different video characteristics. The RS-CNN decides CU partition, depending on the learned features that allow the network to handle different content in video frames, optimizing the process of partitioning and minimizing the unwanted complexity. The SELU activation function has a self-normalizing property that leads to a more stable and faster training process while minimizing the overall complexity. Initially, the blocks are partitioned by using QT based partitioning method. The partition of CU is performed using the RS-CNN with SELU method which minimizes overall complexity and quickens the process of encoding. Fig. 1 represents the process of CU partition using RS-CNN with SELU method.

3.1 Dataset

The datasets used for the video encoding are Joint Collaborative Team on Video Coding (JCT-VC) [23] and Ultra Video Group (UVG) [24]. These datasets include video frames and which are majorly used for video coding or compression. The detailed explanation of these datasets is explained below.

3.1.1. JCT-VC dataset

The JCT-VC dataset is the group of video coding experts from ITU-T study group 16 (VCEG) and the ISO/IEC JTC 1/SC 29/WG 11 (MPEG) developed in 2010 for creating the new generation standard of video coding. This process minimizes the rate of data required for high-quality video coding by approximately 50%. This dataset includes 34 videos that are separated into 14 videos of training, 6 videos of validation, and 14 videos of testing. The training and testing videos are chosen randomly from various resolutions to ensure effectiveness and assess its performance. Fig. 2 represents the sample images of JCT-VC dataset.

Figure. 2 Sample images in JCT-VC dataset

Figure. 3 Sample images in UVG dataset

3.1.2. UVG dataset

The UVG dataset includes 7, 1080p video sequences with high frame rates. Moreover, it contains 16 versatile 4k (3840×2160) testing video sequences captured at 50/120 fps. Fig. 3 represents the sample images of UVG dataset.

3.2 CU partition

The data is given as input to the phase of the CU partition which separates the video frames into various partitions for video coding. The major aim of utilizing the QuadTree (QT) based partitioning in video coding is that it provides efficient coding through including the concept of Prediction Units (PU) and Transform Units (TU) quadtree. The video frame is partitioned to CTUs with 64×64 block size that is same as the Macro Blocks (MB) of 16×16. The CTU is partitioned into various CUs that have size ranges from 8×8 to 64×64. The large size gives matching of HD video content which enhances HEVC coding effectiveness. The QT allows flexibility in the CTU partitioning, even when the size of CTU is higher than 16×16. The CU includes whole CTU or it is partitioned into 4 sub-CUs which are similar in size and represented as leaf nodes. The "1" describes that the CU is partitioned further, while the "0" describes that the CU is not partitioned further. The CU includes many PU and TU keeps data in prediction technique (Inter or Intra-prediction) which is deployed for block. The TU contains a probable, minimum size of 4×4 and offers an adaptive flexibility required in various applications of encoders or decoders. Fig. 4 represents the process of CU partition using Quad Tree.

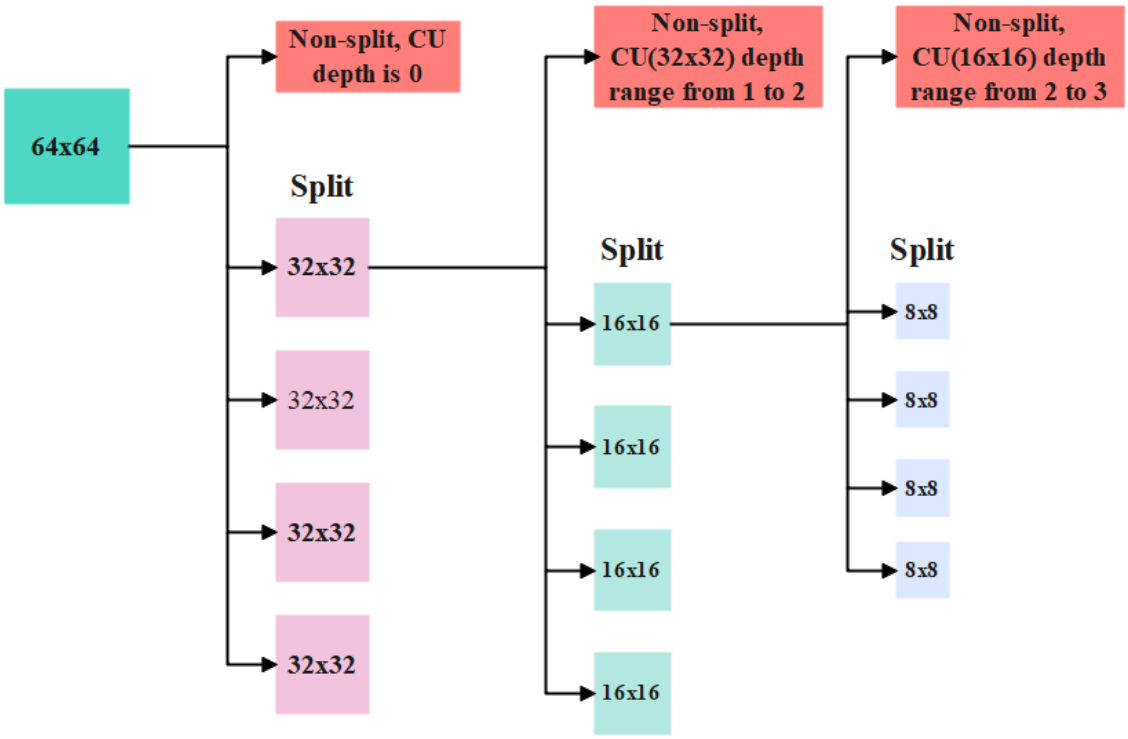

Figure. 4 Process of Quad Tree for CU partition

3.3 Refined stacked-convolutional neural network (RS-CNN)

The RS-CNN method enhances the performance of CNN by stacking multiple layers or networks in a refined manner. The architecture of RS-CNN involves stacking multiple CNNs, wherein every layer concentrates on various features of input data. By stacking the multiple layers in CNN, the RS-CNN method captures more complex patterns and enhances the method's performance for intra-prediction of CU partition. The RS-CNN method learns much more complicated and different sets of image features through training of the diverse CNN methods with varied architectures and hyperparameters. The refining in S-CNN architecture represents the optimum configurations in architecture by varying the depth, width and layers used. This optimized training strategy and architecture results in quick convergence and reduced time, alongside computational resources required for training. In the end, the image is processed through CNN in stack, after which its outputs are integrated and given to a fully connected layer for final classification.

3.3.1. Convolution layer

The initial 64×64 matrix of pixels is provided as input to the convolutional layer. In this research, size of convolution kernel is fixed to the same as step of this layer. To understand the deep features, a convolution block is developed and set in convolution path $C_{i-j}(i = 1,2,3; j = 3,4)$ and C_{3-2}. Initially, input blocks are convoluted through a principal and later, the branch paths obtain 2 feature maps of $Con - 1 \times 1$ and $Con - 2 \times 2$. Next, the convolution blocks and feature maps from 2 various paths are stacked in channel dimension to allow the network learn various mixtures of CU sizes. The main convolution kernels and side branch are 1×1 and 2×2. The convolutions step size are set to 1 which ensures that the feature map has a similar size to facilitate the feature fusion. These features have high texture data to learn CU partition. The convolution kernels of $C_{i-1}(i = 1,2,3)$ are set to half the size of present CU for attaining global data of image. These features are nearly relevance to present CU partition. At last, for fusing these feature maps of two paths, $C_{i-4}(i = 1,2,3)$ and $C_{i-1}(i = 1,2,3)$ channels are set to 64 and 128. Additionally, the non-overlapping of 2×2 max pooling, execution with step size 2 is utilized for filtering features in a major channel $C_{i-j}(i = 1,2,3; j = 2,3,4)$. This minimizes count of parameters and over-fitting risk when protecting the significant features.

3.3.2. Max pooling

This layer is within the Conv2D that supports in the sampling of feature map and minimizes the spatial dimension of input features. The size of 2×2 is utilized for stacked CNN methods with the highest value in every pooling window selected for the new feature map.

3.3.3. Hidden layer and dropout regularization

In this research, various number of hidden layers are set for each CNN method after flattening the images to generate more accurate and efficient output. Here, the set of hidden layers is monitored because the method experiences underfitting when the number of hidden layers is too low. In this research, Adaptive Moment Estimation (Adam) optimizer is utilized to automatically update model parameters based on executing the exponential moving gradient mean.

3.3.4. SELU activation function

In the SELU activation function, the neuron activations automatically converge towards 0 mean and 1 variance. In CU partitioning, the self-normalization maintains consistent and stable feature representation in various layers that results in good decision-making in partitioning. Then, the gradient flows are also improved for much correct learning of complex patterns in video frames that determine the optimal CU partitions. The mathematical formulation for the SELU activation function is given in Eq. (1).

$$selu(x) = \lambda \begin{cases} x, & if x > 0 \\ \alpha e^x - \alpha, & if x \leq 0 \end{cases} \quad (1)$$

In the above Eq. (1), the value of $\alpha \approx 1.6733$ and $\lambda \approx 1.0507$. The values of α and λ are attained by resolving the fixed-point equations to provide an activation function. The below characteristics ensure the property of self-normalizing.

- The values of positive and negative are used to control the mean.
- The saturation regions dampen the difference if it is too high in lower layer.
- The slope is higher than 1 for maximizing the variance when it is too small in the lower layer.

By utilizing the SELU activation function, neuron activations are pushed towards 0 mean and 1 variance. The cross-entropy error is utilized as the loss function and mathematical formulation is given in Eq. (2).

$$crossEntropy = -(i\log(p) + (1-i)\log(1-p)) \quad (2)$$

In the above Eq. (2), i represents the class, log represents the natural algorithm, and p represents the probability for prediction.

3.3.5. Fully connected layer

The connection layers of 3 1D vectors are utilized as inputs for fully connected layers. The possibilities of CU partition of various sizes are acquired through three phases of RS-CNN results, which are finally respective to 21 decision results and output layer is activated through SELU activation function. Moreover, value of QP is included into a fully connected layer as the external feature, so that partition of CU is effectively adapted to various QP values. Then, the mechanism of early termination is utilized to minimize redundancy CU division coding process. The input has local features gathered from different layers for every class. Additionally, QP is essential in process of CU partition, which is spliced to a fully connected layer as supplementary features. These features are merged and combined in this layer until all features learn rules of CTU partition completely, and next the results are utilized to judge the present CU.

3.3.6. Model training

The data samples utilized for training the RS-CNN with SELU are organized by CU size. All CUs that are separated are chosen and extracted from every CU as input, and type of division of CU is mapped and categorized by QP values. In the training process, utilizing the grid search for training every independent CU, alongside composing the training of RS-CNN with SELU is carried out to finish the training process.

Initially, data with a QP value of 22 is chosen for training and training parameter is set to 10 to achieve best prediction. The QP is an essential parameter in video coding which controls the trade-off between compression efficiency and video quality. The lower QP values result in high quality and high QP values giving lesser quality. This value is chosen to represent high-quality encoding, ensuring that the video quality remains close to the actual quality after reaching a certain compression level. Utilizing the range of QP values of 22, the method learns to handle various levels of compression. This research requires 2 QP datasets of JCT-VC and UVG for training the RS-CNN with SELU. Then, the threshold is set to 0.9 for binarizing the result sequence of RS-CNN with SELU to avoid misclassification and enhance accuracy of the CU partition. Therefore, RS-CNN with SELU method has an effective classification performance and is applied to decide on type of CU partition for quickening the process of CU partition. By correctly predicting the optimum CU partitions, the RS-CNN with the SELU method removes all possible partition combinations. This process effectively minimizes the number of executions needed in the encoding process.

4. Experimental results

The RS-CNN with SELU method is simulated with MATLAB 2020a environment and the required system configurations are i5 processor, windows 10 OS and 16GB RAM. The performance metrics used for evaluating the RS-CNN with the SELU method's performance are Bjontegaard Delta Bit Rate (BD-BR) and Bjontegaard Delta Peak-to-Signal Noise Ratio (BD-PSNR), with percentage of coding time (ΔT) and Multi Scale – Structural Similarity Index Measure (MS-SSIM). The mathematical formulation for respective performance metrics is given from Eqs. (3) to (6).

$$BD-BR = \frac{1}{\Delta D}\int_{D_l}^{D_h}\left(r_{proposed} - r_{actual}\right) dD \quad (3)$$

In the above Eq. (3), $\Delta D = D_h - D_l$, the D_h and D_l represent the high and low ends of RD curve range. Also, $r_{proposed}$ and r_{actual} denote the respective bit rates of RS-CNN with SELU and the actual method.

$$BD-PSNR = \frac{1}{\Delta r}\int_{r_l}^{r_h}\left(D_{proposed}(r) - D_{actual}(r)\right) dr \quad (4)$$

In the above Eq. (4), $\Delta r = \log(R_h) - \log(R_l)$, $r_h = \log(R_h)$ and $r_l = \log(R_l)$, representing output range of high and low bitrates of the RD curve. The $D_{proposed}(r)$ and $D_{actual}(r)$ signify RD curves respective to RS-CNN with SELU method and actual method.

$$\Delta T = \frac{T_{proposed} - T_{actual}}{T_{actual}} \times 100 \quad (5)$$

In the above Eq. (5), T_{actual} and $T_{proposed}$ are coding times consumed by the original and RS-CNN with SELU method.

$$MS-SSIM = \frac{1}{M}\sum_{j=1}^{M} SSIM\left(x_j, y_j\right) \quad (6)$$

4.1 Analysis of JCT-VC dataset

Table 1 represents the class distribution of the JCT-VC dataset with different sequences and resolution sizes. The sequences in the JCT-VC dataset are separated into different classes with different resolution sizes for every class. The detailed explanation of sequences and resolution size distributions are explained in Table 1.

Table 2 presents the performance of RS-CNN with SELU method on the JCT-VC dataset with different classes from A to D. The performance metrics of BD-BR, BD-PSNR, ΔT and MS-SSIM for different classes and the average for performance metrics are evaluated. The RS-CNN with SELU method attains an average BD-BR of 2.569%, average BD-PSNR of -0.117 dB, average ΔT of -94.472%, and -53.51% of average MS-SSIM.

Table 3 presents the performance of RS-CNN with the SELU method on the JCT-VC dataset with different existing methods. The different existing methods considered to evaluate the RS-CNN with the SELU method are Multi-Layer Perceptron (MLP), Recurrent Neural Network (RNN), Long-Short Term Memory (LSTM) and traditional CNN. The RS-CNN with SELU method accomplishes BD-BR of 2.569%, BD-PSNR of -0.117 dB, ΔT of -94.472%, and -53.51% of MS-SSIM. By using the refined stacking layers in the CNN method, the network learns complete feature representation essential for the accurate CU partitioning. The SELU activation function is deployed in the training process of RS-CNN that maintain stable activations, leading to quick convergence.

4.2 Analysis of UVG dataset

Table 4 displays the performance of RS-CNN with SELU method on the UVG dataset with different classes of Beauty, Jockey, CityAlley, FlowerFocus and SunBath. The performance metrics namely, BD-BR, BD-PSNR, ΔT and MS-SSIM for different classes and the average for performance metrics are evaluated. The RS-CNN with SELU method achieves an average BD-BR of 3.320%, average BD-PSNR of -0.302 dB, average ΔT of -95.682%, and -54.78% of average MS-SSIM.

Table 1. Class distribution

Classes	Sequence	Resolution Size
A	PeopleOnStreet, Traffic	2560×1600
B	BasketballDrive, BQTerrace, Cactus	1920×1024
C	BasketballDrill, BQMall, PartyScene	833×488
D	BlowingBubbles, BQSquare, RaceHorses	384×192

Table 2. Performance of RS-CNN with SELU method in terms of classes on JCT-VC dataset

Classes	BD-BR (%)	BD-PSNR (dB)	ΔT (%)	MS-SSIM (%)
A	1.895	-0.085	-95.145	-54.67
B	2.461	-0.096	-94.760	-53.71
C	2.893	-0.104	-94.148	-53.49
D	3.029	-0.183	-93.836	-52.18
Average	2.569	-0.117	-94.472	-53.51

Table 3. Performance of RS-CNN with SELU method with different methods on JCT-VC dataset

Methods	BD-BR (%)	BD-PSNR (dB)	ΔT (%)	MS-SSIM (%)
MLP	4.036	-0.319	-93.022	-56.09
RNN	3.761	-0.268	-93.381	-55.42
LSTM	3.156	-0.234	-93.859	-54.84
CNN	2.903	-0.194	-94.012	-54.27
RS-CNN with SELU	2.569	-0.117	-94.472	-53.51

Table 4. Performance of RS-CNN with SELU method in terms of classes on UVG dataset

Classes	BD-BR (%)	BD-PSNR (dB)	ΔT (%)	MS-SSIM (%)
Beauty	3.910	-0.384	-94.234	-53.27
Jockey	3.774	-0.326	-95.459	-54.89
CityAlley	3.248	-0.299	-95.703	-54.73
FlowerFocus	2.998	-0.268	-96.237	-55.38
SunBath	2.673	-0.234	-96.781	-55.67
Average	3.320	-0.302	-95.682	-54.78

Table 5. Performance of RS-CNN with SELU method with different methods on UVG dataset

Methods	BD-BR (%)	BD-PSNR (dB)	ΔT (%)	MS-SSIM (%)
MLP	4.493	-0.470	-91.16	-56.03
RNN	4.172	-0.448	-92.92	-55.79
LSTM	3.971	-0.391	-93.48	-55.23
CNN	3.782	-0.348	-94.09	-54.92
RS-CNN with SELU	3.320	-0.302	-95.68	-54.78

Table 5 displays the performance of RS-CNN with SELU method on the UVG dataset with different existing methods. The different existing methods considered to evaluate the RS-CNN with SELU method are MLP, RNN, LSTM and traditional CNN. The RS-CNN with SELU method achieves BD-BR of 3.320%, BD-PSNR of -0.302 dB, ΔT of -95.68%, and -54.78% of MS-SSIM.

4.3 Analysis of encoding time

In this section, the encoding time of RS-CNN with SELU method is analyzed for both datasets with different resolution sizes. The different existing methods considered to evaluate the RS-CNN with SELU method are MLP, RNN, LSTM and traditional CNN. Then, the different resolution sizes considered are 2560×1600, 1920×1024, 833×488 and 384×192.

The RS-CNN with SELU method effectively learns the complex patterns in video sequences, which allows more informed decisions about CU partitioning when compared to the existing methods. By correctly predicting the optimum CU partitions, the RS-CNN with the SELU method removes all possible partition combinations. This process effectively minimizes the number of executions needed in the encoding process.

The RS-CNN with SELU method concentrates exceptional resources on optimal CU partitions, which minimizes the overall complexity and quickens the process of encoding. Table 6 presents the performance of RS-CNN with SELU method with different methods on encoding time.

4.4 Comparative analysis

In this section, the RS-CNN with SELU method is compared to the existing methods of Hybrid method [18] and D-CNN [19] on JCT-VC dataset, TCVC-Net [20] and DCVC [21] on UVG dataset. In Table 7, the RS-CNN with SELU method is compared to the JCT-VC dataset and in Table 8, it is compared to the UVG dataset and MCL-JCV dataset.

In table 7, the RS-CNN with SELU method achieves less BD-BR of 1.83% on the PeopleOnStreet sequence in JCT-VC dataset and 1.96% on the Traffic sequence in JCT-VC dataset.

Table 6. Performance of RS-CNN with SELU method with different methods on encoding time

Methods	Resolution size	Encoding time (s)	
		JCT-VC	UVG
MLP	2560 × 1600	407.24	434.89
	1920 × 1024	398.61	409.21
	833 × 488	344.79	386.48
	384 × 192	317.23	351.59
RNN	2560 × 1600	457.82	461.48
	1920 × 1024	418.34	426.70
	833 × 488	376.90	391.27
	384 × 192	352.16	365.94
LSTM	2560 × 1600	376.98	407.69
	1920 × 1024	341.39	379.27
	833 × 488	302.73	341.84
	384 × 192	297.03	308.39
CNN	2560 × 1600	312.87	361.47
	1920 × 1024	284.56	328.05
	833 × 488	241.49	298.56
	384 × 192	218.05	258.20
RS-CNN with SELU	2560 × 1600	247.31	276.39
	1920 × 1024	201.93	231.25
	833 × 488	179.37	194.30
	384 × 192	142.80	153.71

Table 7. Comparative analysis on JCT-VC dataset

Sequences	Methods	BD-BR (%)	BD-PSNR (dB)	ΔT (%)
PeopleOnStreet	Hybrid method [18]	4.48	NA	-96.23
	D-CNN [19]	1.99	-0.11	NA
	Proposed RS-CNN with SELU	1.83	-0.08	-96.57
Traffic	Hybrid method [18]	4.18	NA	NA
	D-CNN [19]	2.12	-0.12	NA
	Proposed RS-CNN with SELU	1.96	-0.09	-93.72

Table 8. Comparative analysis

Methods	Methods	BD-BR (%)	BD-PSNR (dB)
UVG	TCVC-Net [20]	-7.836	0.227
	DCVC [21]	NA	-46.33
	Proposed RS-CNN with SELU	3.320	-0.302
MCL-JCV	TCVC-Net [20]	1.254	-0.045
	DCVC [21]	NA	-37.07
	Proposed RS-CNN with SELU	1.134	-41.27

In Table 8, the RS-CNN with SELU method achieves BD-BR of 3.320% on the UVG dataset, 1.134% on MCL-JCV dataset. The RS-CNN with SELU method includes multiple CNN layers stacked together that enhance the ability to learn and predict CU partition. The SELU activation function maintains stable activations during the training process of the RS-CNN method, effectuating faster convergence in the CU partition.

4.5 Discussion

The results of the RS-CNN with the SELU method are analyzed with two datasets, JCT-VC and UVG, in terms of BD-BR, BD-PSNR, ΔT and MS-SSIM. The performance of RS-CNN with SELU method is compared with the MLP, RNN, LSTM and traditional CNN. Moreover, the RS-CNN method is compared with the existing algorithms like Hybrid method [18] and D-CNN [19] on JCT-VC dataset, TCVC-Net [20] and DCVC [21] on the UVG dataset. These existing algorithms have drawbacks of high complexity and higher time consumption for the encoding process, and less prediction accuracy. By using the refined stacking layers in the CNN method, it learns complete feature representation which is essential for the correct partitioning of CU. The SELU activation function is employed in the training process of RS-CNN that maintains stable activations and culminating in quicker convergence for accurate CU partition. The RS-CNN with SELU method is proposed in this research to minimize the complexity and reduce the time consumption of the encoding process in the CU partition. The RS-CNN with SELU method accomplishes BD-BR of 2.569% on the JCT-VC dataset and 3.320% of BD-BR on the UVG dataset.

5. Conclusion

The RS-CNN with SELU method is proposed in this research to minimize the complexity and time consumption of the encoding process in the CU partition. The partition of CU is performed by using the RS-CNN with the SELU method, which minimizes the overall complexity and quickens the process of encoding. By using the refined stacking layers in the CNN method, it learns complete feature representation which is essential for the correct partitioning of CU. The SELU activation function is used in the training process of RS-CNN that maintains stable activations and leads to quicker convergence for accurate CU partition. By correctly predicting the optimum CU partitions, the RS-CNN with the SELU method removes all possible partition combinations. This process effectively minimizes the number of executions needed in the encoding process. The RS-CNN with SELU method accomplishes BD-BR of 2.569% on the JCT-VC dataset and 3.320% of BD-BR on the UVG dataset. In future, different DL based algorithms can be used for CU partition to further improve the performance of the prediction process.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

The paper conceptualization, methodology, software, validation, formal analysis, investigation, resources, data curation, writing—original draft preparation, writing—review and editing, visualization, have been done by 1st author. The supervision and project administration, have been done by 2nd author.

References

[1] D. García-Lucas, G. Cebrián-Márquez, A.J. Díaz-Honrubia, T. Mallikarachchi, and P. Cuenca, "A fast full partitioning algorithm for HEVC-to-VVC video transcoding using Bayesian classifiers", *Journal of Visual Communication and Image Representation*, Vol. 94, p. 103829, 2023.

[2] J. Sainio, A. Mercat, and J. Vanne, "RDO Candidate Selection for Maximizing Coding Efficiency in a Practical HEVC Encoder", In: *Proc. of ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Rhodes Island, Greece, pp. 1-5, 2023.

[3] Z. Sun, L. Yu, and W. Peng, "QTMT-LNN: A fast intra CU partition using lightweight neural network for 360-degree video coding on VVC", *IET Image Processing*, Vol. 17, No. 2, pp. 597-612, 2023.

[4] L. Chen, B. Cheng, H. Zhu, H. Qin, L. Deng, and L. Luo, "Fast Versatile Video Coding (VVC) Intra Coding for Power-Constrained Applications", *Electronics*, Vol. 13, p. 2150, 2024. https://doi.org/10.3390/electronics13112150

[5] N.V. Thang, "Hierarchical random access coding for deep neural video compression", *IEEE Access*, Vol. 11, pp. 57494-57502, 2023.

[6] P. Hari, and K.N.S. Batta, "High-Speed Coding Unit Depth Identification Based on Texture Image Information Using SVM", *Defence Science Journal*, Vol. 74, No. 2, pp. 270-277, 2024.

[7] R. Kumar, K. Kumar, S. Mahajan, and A.K. Pandit, "Study and implementation of K-multiple constraint shortest path for H. 265 HEVC for optimal video compression", *Journal of Ambient Intelligence and Humanized Computing*, Vol. 14, No. 5, pp. 6149-6164, 2023.

[8] A. Meyer, F. Brand, and A. Kaup, "Learned wavelet video coding using motion compensated temporal filtering", *IEEE Access*, Vol. 11, pp. 113390-113401, 2023.

[9] J. Chen, M. Wang, P. Zhang, S. Wang, and S. Wang, "Sparse-to-Dense: High Efficiency Rate Control for End-to-end Scale-Adaptive Video Coding", *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 34, No. 5, pp. 4027-4039, 2024.

[10] S. Chen, S. Aramvith, and Y. Miyanaga, "Learning-Based Rate Control for High Efficiency Video Coding", *Sensors*, Vol. 23, p. 3607, 2023.

[11] J. Wang, X. Shang, X. Zhao, and Y. Zhang, "A convolutional neural network-based rate control algorithm for VVC intra coding", *Displays*, Vol. 82, p. 102652, 2024.

[12] S.K. Sairam, and P. Muralidhar, "A Deep Learning Approach in Scalable High Efficiency Video Coding for Fast Coding Unit Size Decision", *IETE Technical Review*, Vol. 40, No. 3, pp. 287-302, 2023.

[13] M. Yang, J. Huo, X. Zhou, W. Qiao, S. Wan, H. Wang, and F. Yang, "Joint Rate-Distortion Optimization for Video Coding and Learning-Based In-Loop Filtering", *IEEE Transactions on Multimedia*, Vol. 26, pp. 2851-2865, 2024.

[14] Y. Bian, X. Sheng, L. Li, and D. Liu, "LSSVC: A Learned Spatially Scalable Video Coding Scheme", *IEEE Transactions on Image Processing*, Vol. 33, pp. 3314-3327, 2024.

[15] J.T. Fang, and J.K. Chen, "Multiple layers complexity allocation with dynamic control scheme for high-efficiency video coding", *Journal of Real-Time Image Processing*, Vol. 21, No. 3, p. 69, 2024.

[16] J. Xu, B. Wang, Q. Peng, and W. Li, "Key-frame reference selection for error resilient video coding using low-delay hierarchical coding structure", *Signal, Image and Video Processing*, Vol. 18, No. 1, pp. 215-222, 2024.

[17] Y.M. Vaidya, and S.P. Metkar, "Computational Complexity Reduction of HEVC by Early Termination of Transform Unit Partitioning Using Raster Scan Approach", *SN Computer Science*, Vol. 5, No. 2, p. 208, 2024.

[18] V. Galiano, H. Migallón, M. Martínez-Rach, O. López-Granado, and M.P. Malumbres, "On the use of deep learning and parallelism techniques to significantly reduce the HEVC intra-coding time", *The Journal of Supercomputing*, Vol. 79, No. 11, pp. 11641-11659, 2023.

[19] T. Wang, G. Wei, H. Li, T. Bui, Q. Zeng, and R. Wang, "A Method to Reduce the Intra-Frame Prediction Complexity of HEVC Based on D-CNN", *Electronics*, Vol. 12, p. 2091, 2023.

[20] D. Jin, J. Lei, B. Peng, Z. Pan, L. Li, and N. Ling, "Learned video compression with efficient

temporal context learning", *IEEE Transactions on Image Processing*, Vol. 32, pp. 3188-3198, 2023.
[21] H. Guo, S. Kwong, D. Ye, and S. Wang, "Enhanced context mining and filtering for learned video compression", *IEEE Transactions on Multimedia*, Vol. 26, pp. 3814-3826, 2024.
[22] Y. Wang, P. Dai, J. Zhao, and Q. Zhang, "Fast CU Partition Decision Algorithm for VVC Intra Coding Using an MET-CNN", *Electronics*, Vol. 11, p. 3090, 2022.
[23] JCT-VC dataset: https://www.itu.int/en/ITU-T/studygroups/2017-2020/16/Pages/video/jctvc.aspx
[24] UVG dataset: https://ultravideo.fi/dataset.html