Hindawi
Scientific Programming
Volume 2020, Article ID 8883214, 11 pages
https://doi.org/10.1155/2020/8883214

Research Article

Adaptive CU Split Decision Based on Deep Learning and Multifeature Fusion for H.266/VVC

Jinchao Zhao, Yihan Wang, and Qiuwen Zhang

College of Computer and Communication Engineering, Zhengzhou University of Light Industry, Zhengzhou 450002, China

Correspondence should be addressed to Qiuwen Zhang; zhangqwen@126.com

Received 10 March 2020; Revised 9 June 2020; Accepted 7 July 2020; Published 1 August 2020

Academic Editor: Fabrizio Riguzzi

With the development of technology, the hardware requirement and expectations of user for visual enjoyment are getting higher and higher. The multitype tree (MTT) architecture is proposed by the Joint Video Experts Team (JVET). Therefore, it is necessary to determine not only coding unit (CU) depth but also its split mode in the H.266/Versatile Video Coding (H.266/VVC). Although H.266/VVC achieves significant coding performance on the basis of H.265/High Efficiency Video Coding (H.265/HEVC), it causes significantly coding complexity and increases coding time, where the most time-consuming part is traversal calculation rate-distortion (RD) of CU. To solve these problems, this paper proposes an adaptive CU split decision method based on deep learning and multifeature fusion. Firstly, we develop a texture classification model based on threshold to recognize complex and homogeneous CU. Secondly, if the complex CUs belong to edge CU, a Convolutional Neural Network (CNN) structure based on multifeature fusion is utilized to classify CU. Otherwise, an adaptive CNN structure is used to classify CUs. Finally, the division of CU is determined by the trained network and the parameters of CU. When the complex CUs are split, the above two CNN schemes can successfully process the training samples and terminate the rate-distortion optimization (RDO) calculation for some CUs. The experimental results indicate that the proposed method reduces the computational complexity and saves 39.39% encoding time, thereby achieving fast encoding in H.266/VVC.

1. Introduction

With higher requirements for video compression, more effective video coding standards have become critical. The JVET has developed a next-generation video coding standard called the H.266/VVC [1]. The H.266/VVC Test Model (VTM) [2] has implemented many novel technologies, which can significantly enhance the coding efficiency. Specifically, the H.266/VVC uses a quad-tree with nested multitype tree (QTMT) coding architecture for CU partition [3], which shows better coding performance, but leads to the extremely coding computational complexity. And the coding complexity may be 5 times that of the H.265/HEVC [4]. Figure 1 shows 67 intraprediction modes, where Planar and DC mode are in line with H.265/HEVC. Therefore, the prediction modes are very dense, which can gain more precise prediction, but the computational complexity is significantly high. In addition, other additional tools are introduced to enhance the coding efficiency, such as Position Dependent Intraprediction Combination (PDPC) [5] and Multiple Transform Selection (MTS) [6]. In short, the novel technologies, which include QTMT partitioning structure, extended intraprediction mode, PDPC, and MTS, significantly enhance performance of H.266/VVC but lead significantly to the computational complexity. The intracoding complexity of the VTM is 18 times that of the H.265/HEVC test Model (HM) in "All Intra" configuration condition [7]. Thus, it is essential for H.266/VVC to exploit a fast coding algorithm which suffices the actual requirements of the potential market.

The H.266/VVC proposes some novel technologies for intracoding based on H.265/HEVC and extends some previous methods, where the key concept in these tools is MTT structure [8]. The flexible block size can achieve excellent coding performance, and the CU partition structure is the core of the coding layer. The QTMT partition structure is used by H.266/VVC, which causes most of the increase in complexity. Specifically, the coding tree unit (CTU) is firstly partitioned by a quad-tree (QT). Afterwards, the quad-leaf node is further divided by MTT structure. There are four partition types in MTT structure, including vertical binary tree partition (BV), horizontal binary tree split (BH), vertical ternary tree split (TV), and horizontal ternary tree split (TH). Therefore, the MTT structure supports more CU shapes than QT, which obtains more efficient encoding performance. Moreover, once the CU is partitioned by MTT structure, QT is no longer valid in the subsequent partition process, which can simplify the CU partition process. The QT and MTT size are restricted by some defined encoding parameters. The CU size is changed from maximum 128×128 to minimum 8×4 or 4×8 based on these parameters and the split type. 128×128 refers to the normal size, and the maximum accepted size is 64×64 in "All Intra" configuration.

Many types of leaf nodes which are called CU are used for prediction and transformation. Therefore, the QTMT structure eliminates the hierarchical structure of CU, PU, and TU. As shown in Figure 2, the solid black line indicates QT partition and the dotted line indicates MT partition in QTMT architecture. As we all know, the H.265/HEVC searches for the best CU partition, which can suffer from the high complexity by using QT structure. Compared to QT structure of H.265/HEVC, the process of obtaining the best CU is more complex in QTMT structure of H.266/VVC. Furthermore, more CU shapes greatly increase the complexity of intraprediction and have longer encoding time but greatly improve the encoding efficiency in H.266/VVC.

The remainder of this paper is organized as follows: Section 2 shows the related work. In Section 3, the proposed fast CU partition method is described in detail, which includes observations and motivations, texture classification model based on threshold, an adaptive CNN structure, and a CNN structure based on multifeature fusion. Section 4 shows the experimental results. Finally, the conclusion is provided in Section 5.

2. Related Work

In view of the above problems, there are some works on CU partition decision of H.266/VVC to reduce the coding complexity. Recently, many fast CU partition methods have been proposed in literatures, which include heuristic and learning-based method. There are heuristic schemes: a fast intracoding method based on H.266/VVC is presented in [9], which combines a low-complexity CTU architecture derivation method and a fast intramode decision method to accelerate running speed. The local constrained CU partition decision methods are illustrated in [10]. An effective QTBT partition decision method is designed in [11] to achieve a good balance between the computational complexity and RD performance. A fast intra-CU partition decision method based on spatial characteristics is introduced in [12], where an adaptive threshold is obtained by adjusting the error rates of OS_BTD2 and ES_BTD3. Several fast intra-algorithms which improve the overall balance between complexity and gain in H.266/VVC are proposed in [13] to reduce the

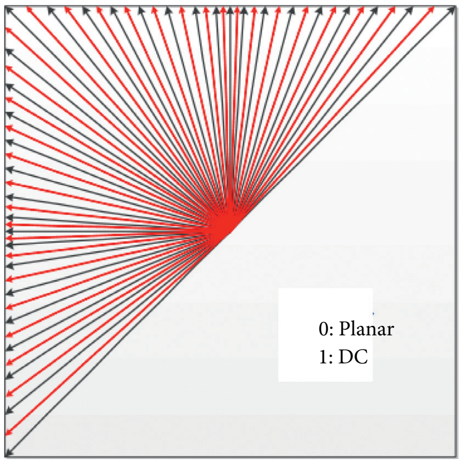


Figure 1: 67 intraprediction modes of VVC.

number of tests. A new fast CU partition algorithm based on Bayesian decision rule is proposed in [14]. Using the average depth information of the adjacent largest coding unit (LCU) is devised in [15] to determine whether the CU split is terminated in advance or not. Moreover, the unnecessary RDO is effectively eliminated by using the coding mode of the adjacent CUs to accelerate encoding in H.266/VVC. A pruning algorithm based on the prediction CU size is introduced in [16] to reduce the redundant MTT partition. A method which can skip the redundant MTT pruning for H.266/VVC is presented in [17]. A fast split algorithm based on variance and gradient is designed in [18] to solve the rectangular spilt problem in H.266/VVC.

Recently, the learning-based methods have attracted more and more attention and have made significant improvements in performance, which are divided into two parts to present. One is the learning-based algorithm in H.265/HEVC. In [19], the CU partition process is modeled as a two-class classification problem for H.265/HEVC. A deep CNN structure is presented in [20] to predict the CTU partition instead of RDO calculation. The deep learning approaches are proposed in [21, 22] to predict the CU partition for reducing the complexity of H.265/HEVC. The other part is that some experts propose algorithms based on deep learning and machine learning (ML) to accelerate the coding process of H.266/VVC. A fast QTBT partition method based on ML technology is introduced in [23], which utilizes random forest classifiers to determine the partition modes of each CU. Fast CU depth decision algorithms based on CNN are proposed in [11, 24] to model the QTBT partition depth range as a multiclass classification problem. A fast intraprediction mode decision algorithm based on ML technology is introduced in [25], which can reduce encoding time. An adaptive CU split decision method based on H.266/VVC is designed in [26], which uses a variable CNN to avoid the calculation of full RD. A lightweight and adjustable QTBT partition scheme based on ML technology is presented in [27], which achieves an adjustable compromise between reduced complexity and reduced video quality in H.266/VVC. A technical overview of a deep-learning-based method aiming at optimizing next-

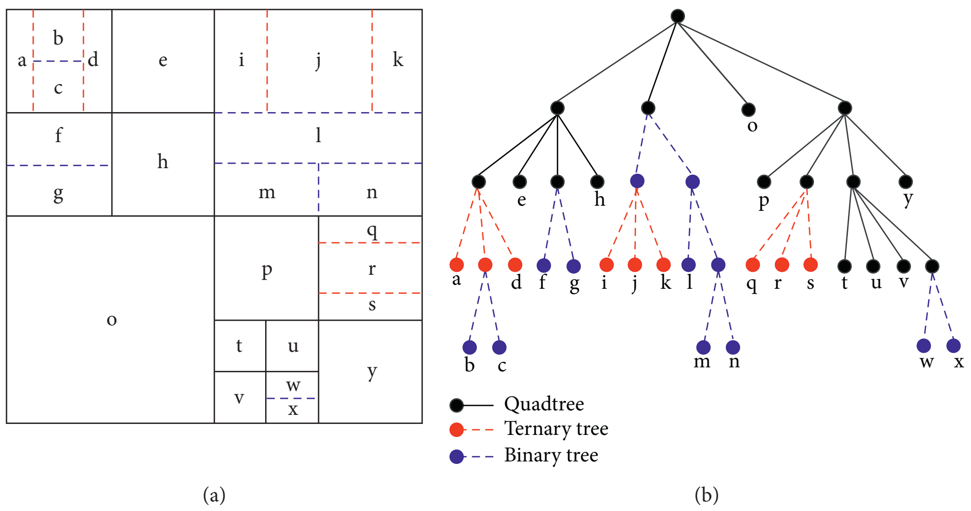


Figure 2: QTMT structure: (a) QTMT partition for CTU; (b) the corresponding tree representation.

generation hybrid video encoders is provided in [28] for driving the CU partition. A CNN oriented fast QTBT partition decision scheme is introduced in [29] for inter-coding. This method analyzes the QTBT in a statistical way, which effectively guides us to design the structure of CNN.

Previous methods on CNN in H.266/VVC are generally for QTBT structure. However, the structure of H.266/VVC has developed into QTMT structure, which makes the high computational complexity. Furthermore, since the depth of CTU has no efficiency and RDO is an operating of time consumption, the depth of CTU and RDO of CU partition decision are difficult in the practical applications. To resolve these issues in the existing algorithms, this paper proposes an adaptive CU split decision method based on deep learning and multifeature fusion. We use multifeature fusion in combination with deep learning to solve the problem of coding complexity. The division depth and shape of CU in QTMT architecture are closely related to the texture complexity. Moreover, the deep learning can better summarize and analyze the data, which utilizes this knowledge to build models to support decisions Firstly, we develop a texture classification model based on threshold to recognize complex and homogeneous CU. Secondly, if the complex CUs belong to edge CU, the CNN structure based on multifeature fusion is used for classification. Otherwise, the adaptive CNN structure is utilized to classify CUs. Finally, the division of CU is determined by the trained network and the parameters of CU. When the CU is split, the above two CNN training schemes can successfully process the training samples and avoid the whole RDO calculation.

3. The Proposed Algorithm

The CNN tools are applied by many applications in the field of video coding. The proposed method introduces an adaptive CU split decision method based on deep learning and multifeature fusion. Firstly, we develop a texture classification model based on CU size, QP, and texture complexity to identify complex and homogeneous CUs in advance by calculating the texture metric. Specifically, if standard deviation (SD) which measures the texture complexity is less than the threshold that is calculated from the texture classification model, the CU is considered as homogeneous CU and is nonsplit. On the contrary, the CU is considered as complex CU. Secondly, if the CUs belong to edge CU, CNN structure based on multifeature fusion is performed to classify CUs. Otherwise, adaptive CNN structure is performed to classify CUs. Finally, according to the above CNN structure inference result, if the current CU is nonsplit and is the smallest CU, the CU division process is termination. Otherwise, the SD and threshold of the sub-CU are recalculated until the end. Figure 3 shows the flowchart of the proposed method based on deep learning and multifeature fusion. In this section, we will mainly show texture classification model based on threshold, the adaptive CNN architecture, and CNN structure based on multifeature fusion.

3.1. Observations and Motivations.

3.1. Observations and Motivations. The previous algorithms always input the original block for intraprediction in CNN structure, because the original blocks contain some features of the block. The concept of intra-subpartition (ISP) is introduced in H.266/VVC, which can divide the intra-CU into several subblocks. The ISP may prevent the original block from correctly displaying the CU split function. In addition, the shapes of CUs are square in H.265/HEVC, while there are both square and rectangle in H.266/VVC. The solutions which have some disadvantages about CU partition in H.265/HEVC cannot be directly used in H.266/VVC. Specifically, the solution is that we use the same size which is obtained using downsampling of all CUs as input in CNN structure, which will lose valuable original information. The other method is that we train numerous CNNs for all kinds of CU size, but these CNNs networks are not efficient.

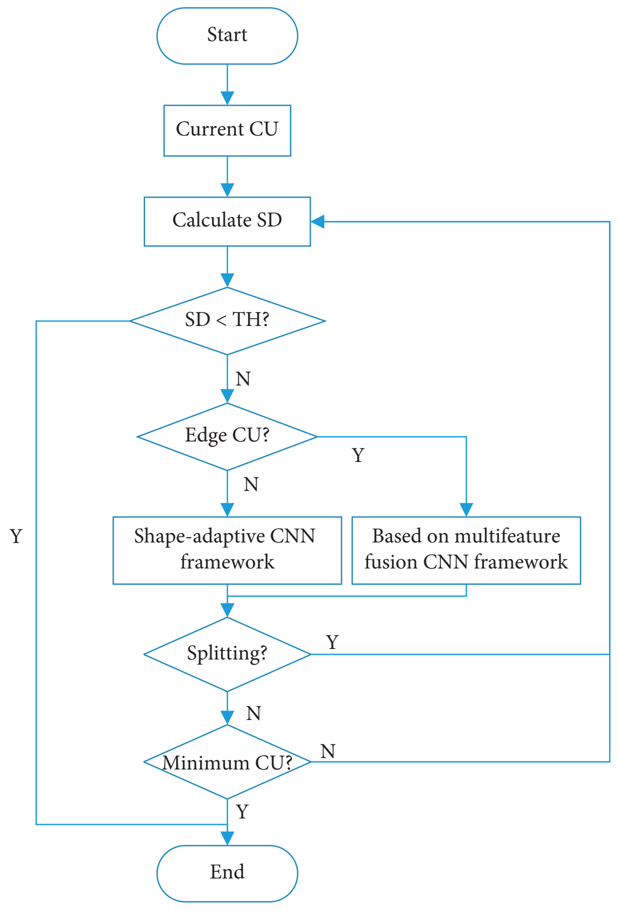


Figure 3: Flowchart of the proposed method.

In addition, the optimal CU size depends on the complexity of adjacent blocks in intracoding of H.266/VVC. Therefore, the pixels of adjacent blocks are significant for high prediction preciseness. In adaptive CNN structure, only the residual blocks of neighboring blocks are used as input. Although the input size is different, the input size will change into the same before the fully connected layer (FCL). Compared with the traditional CNN classifiers with similar network structure, the proposed CNN classifier is designed, which can perform better classification for all kinds of CU size in H.266/VVC. Moreover, we have also designed a CNN structure based on multifeature fusion for the case that the edge CU may be misclassified, which utilizes the texture complexity and depth feature of edge CUs as input, thereby improving the accuracy of classification. The CNN classifier is used to classify CUs, which may not be essential for CUs in homogeneous regions. Thus, before performing adaptive CNN structure and CNN structure based on multifeature fusion, we use heuristic method to avoid unnecessary calculation, which can obtain good efficiency.

Based on the above observation and motivations analysis, a texture classification model based on threshold is developed to recognize CUs in homogeneous region, and the homogeneous regions are determined to be nonsplit. To further reduce the coding complexity, an adaptive CNN structure and CNN structure based on multifeature fusion are designed and trained for CUs in complex regions. Moreover, the loss function (LF) is used to improve the classification accuracy, thereby improving the coding property.

3.2. Texture Classification Model Based on Threshold.

3.2. Texture Classification Model Based on Threshold. The texture classification model based on threshold is developed to enhance split precision in this paper, which has no additional complexity. We calculate the texture complexity for CUs by utilizing SD and perform a large number of experiments. Then, a threshold model which can improve split accuracy is established based on the empirical function of QP and depth.

In the encoding process, the larger CUs are used to the single area of image content. In contrast, the smaller CUs are used to areas with rich details. This phenomenon can motivate us to use the texture complexity of the current CU to determine whether CU is directly split or skipped, and the variance of CU indicates the degree of energy dispersion between the two pixels of CU. Therefore, we classify CUs according to the texture complexity. In addition, we use SD to calculate the texture complexity according to the method of calculating texture complexity in [30], which is calculated as follows:

$$S\ D = \sqrt{\frac{1}{W\times H}\sum_{x=0}^{W-1}\sum_{y=0}^{H-1} p(x,y)^2 - \left(\frac{1}{W\times H}\sum_{x=0}^{W-1}\sum_{y=0}^{H-1} p(x,y)^2\right)^2}, \tag{1}$$

where $W$and H represent the width and height of CU, respectively. $p(x, y)$ represents the pixel value at (x, y).

According to a lot of simulation experiments, we discover that the threshold is related to depth and QP of CU. The best CU depth distribution in four different sequences is calculated in experiments. We find that the CUs at each depth do not exceed about 40% and the percentage of CU is smaller when the depth value is smaller. According to the statistical results, we consider that the CU depth should be applied to the threshold model and should record the effect of depth in the threshold model.

In different types of frame, the depth distribution of CU is different. The calculation method of the depth is defined as

$$P_{\text{CU\_Depth}} = H_{\text{CU\_Depth}} \times W_{\text{CU\_Depth}}, \tag{2}$$

$$P_{\text{Frame}} = H_{\text{Frame}} \times W_{\text{Frame}}, \tag{3}$$

$$R_{\text{CU\_Depth}} = \frac{P_{\text{CU\_Depth}} \times N_{\text{CU\_Depth}}}{P_{\text{Frame}}} \times 100\%, \tag{4}$$

$$E_{\text{Depth}} = \sum_{\text{Depth}=0}^{4} \text{Depth} \times R_{\text{CU\_Depth}}, \tag{5}$$

where W, H, and Depth represent width, height, and depth of CU, respectively. $P_{\text{CU\_Depth}}$ and P_{Frame} denote pixel in depth level and in frame, respectively. E_{Depth} represents the CU depth expectation of different frame types.

We can visualize the CU coded video sequences. The CU partition in a certain frame of *FourPeople* video sequence is shown in Figure 4. According to a large number of experiments, we conclude that the CU size will increase as the QP increases. When QP value is small, the probability of smaller CU is high to avoid larger distortion. Furthermore, other video sequences also have similar phenomenon. Based on the above analysis, it concludes that QP, similar to CU depth, is also an essential element in the texture classification model based on threshold.

Figure 5 demonstrates the correlation in QP, CU depth, and threshold. It is observed that the threshold decreases as QP and CU depth decrease, which can further validate the above analysis. In order to increase the probability of determining the texture region, the threshold will also become smaller when QP becomes smaller. Therefore, an empirical texture classification model based on threshold is established in this paper, where the threshold is a function modeled by the empirical function of QP and CU depth. The threshold is defined as

$$\mathrm{Th} = F(\mathrm{QP}) \times G(\mathrm{Depth}), \tag{6}$$

$$F(\mathrm{QP}) = \frac{\mathrm{QP}}{\sqrt{100 - 1.7 \times \mathrm{QP}}},$$
$$G(\mathrm{Depth}) = \sum_{Depth=0}^{3} \mathrm{Depth} \times R_{\mathrm{CU\_Depth}}, \tag{7}$$

where Th denotes the threshold of texture classification. Depth denotes the depth of the current CU. $F(\mathrm{QP})$ and $G(\mathrm{Depth})$ represent the empirical function. Generally, the threshold depends on two empirical functions and it is hard to make the rigorous proof. Therefore, we change the parameters in two empirical functions and observe the performance. Finally, we gain the threshold.

We compare the calculated SD with threshold in the proposed algorithm. If SD is less than the threshold, the CU belongs to homogeneous area and no longer split. Otherwise, the proposed adaptive CNN classifier and CNN classifier based on multifeature fusion are utilized to classify the complex CUs. In our series of comprehensive tests, the CUs in homogeneous region are identified through the threshold-based texture classification model to skip division. The proposed method does not need to save the external time of CNN, because a part of CU is determined as nonsplit by heuristic method.

3.3. Adaptive CNN Structure. The QTMT partition structure which gets CUs with square and rectangular is one of effective tools in H.266/VVC. In this case, embedding CNN into the encoding step will cause problems because of various sizes in H.266/VVC. Since the general input is a fixed-size CU in the previous research on H.265/HEVC such as [21], the method of downsampling for the input block causes a lot of valuable classification information to be lost. Obviously, this method does not have the optimal solution to solve the problem of various sizes in H.266/VVC. The other method is that we train numerous CNNs for all kinds

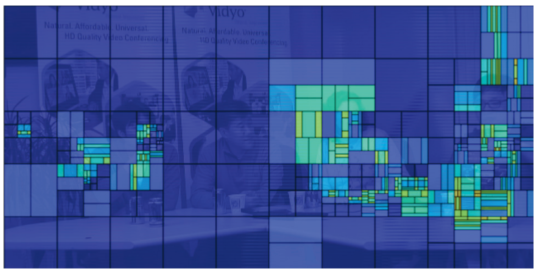


FIGURE 4: The CTU partition in *FourPeople*.

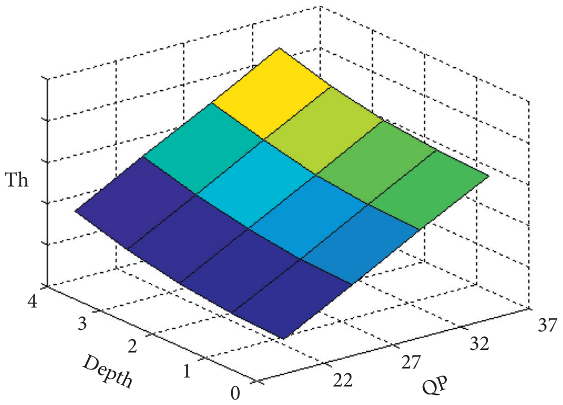


FIGURE 5: The correlation in QP, depth, and threshold.

of CU sizes, but the efficiency of these CNNs network is not high. Therefore, this is also not the best method to settle this problem.

In order to solve the above issues, we propose an adaptive CNN structure, in which the max pooling layer that is one of the effective tools of CNN is customized. Due to the characteristics of adaptive CNN structure, the multishape CUs are processed by only trained CNN and do not need different CNN networks to handle different CU sizes, which greatly improves the network utilization. At the same time, the input of CNN is also adjusted appropriately, where we use the residual of neighboring blocks of the current CU as the input of CNNs. Figure 6 shows the locations of neighboring blocks. In addition, the optimal CU size depends on the complexity of neighboring blocks in intracoding of H.266/VVC. Therefore, the pixels of adjacent blocks are important element for high prediction precision. Before CU split in intraprediction, the trained CNN models are embedded into the encoder. The training time of CNN model is not counted in the total coding time. Therefore, the overhead generated by CNNs has no effect on the total coding time. Furthermore, compared to the conventional encoding process, the proposed method cannot cause the redundant iterations for RDO, which can decrease the computational complexity of intracoding.

In order to determine the most suitable multi-input CNN structure, we evaluate the prediction accuracy based on the difference in the number of inputs, where Class A and Class B sequences are used to evaluate the number of adjacent blocks and the number of parameters (Conv, kernel, and FCL), as

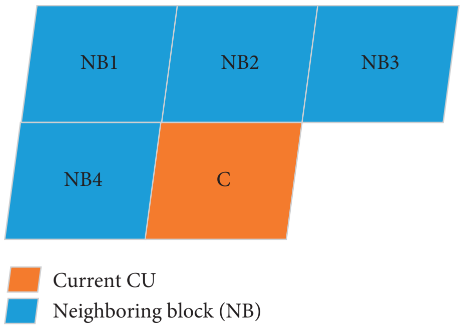


Figure 6: Neighboring block of the current block.

shown in Table 1. It can be seen from Table 1 that the increase of the number of adjacent blocks and kernels leads to improving the prediction accuracy. On the other hand, compared with the kernel, the accuracy of CNN does not fluctuate significantly for the adjustment of the number of Conv and FCLs. Therefore, the number of convolutional layers, kernels, and FCLs will affect the prediction accuracy, where the kernel is an effective parameter for extraction feature map.

According to the evaluation, we select the best performance conditions for CNN model in the proposed method. Therefore, the proposed adaptive CNN method, which consists of two convolutional layers, two FCLs, and ten kernels, utilizes the residual blocks of neighboring blocks of the current CU as input, as shown in Figure 7.

Input layer: the input of adaptive CNN structure is the residual blocks of neighboring blocks of the current CU to maintain valuable original characteristics.

Convolutional layers and max pooling layer: the first layer is convolutional layer with ten kernels. The kernel in this layer is considered a feature extractor. The second layer and fourth layer are the adaptive pooling layer that is designed based on the problem of CU shape in H.266/VVC, where two adaptive pooling layers are used for CUs with width or height greater than 32 and 16, respectively. The pooling layer is used in CNN transmission, which can retain useful information to gain the same size before FCL. The size of the adaptive pooling layer is various by size of residual blocks of the input to maintain valuable original features. Similarly, the third and fourth layer implement convolution layer and adaptive pooling layer.

Fully connected layer: after convolutional layer and pooling layer, all input blocks will be converted into 4×4 block and then enter the FCL. Each neuron in the FCL connects with all neurons in the previous layer. FCL can incorporate specific local information in convolutional layer or pooling layer. Furthermore, the width, height, and QP of CU are added as neuron of FCL, because the width, height, and QP will affect the final CU split decision, which can obtain better classification accuracy.

Activation function: in order to improve the performance of the adaptive CNN network, the rectified linear units (ReLU) function is generally used as the activation function of each neuron in FCLs. It is worth mentioning that all convolutional layers and hidden FCLs are activated with ReLU.

Loss function: for a specific classification task, we need to choose a suitable LF to improve the accuracy of classification. Because QP is a significant factor in the above simulation tests, the adaptive CNN structure employs the cross-entropy LF based on QP. Therefore, the penalty can be adjusted by using QP, and LF will make CNN structure more accurate. Based on the above analysis, the classification result of the current CU is obtained from the information of the neighboring blocks of the current CU, and the LF is defined as

$$J_l = -\frac{1}{m}\sum_{i=1}^{m}\left[\rho_1 \times s_l^i \log\left(\widehat{y_l^i}\right) + \rho_2 \times \left(1 - s_l^i\right)\log\left(1 - \widehat{s_l^i}\right)\right], \tag{8}$$

$$\begin{aligned}\rho_1 &= \frac{\mathrm{QP}}{\sqrt{\mathrm{QP}+800}},\\ \rho_2 &= \frac{1}{\rho_1},\end{aligned} \tag{9}$$

where s_l^i and $\widehat{s_l^i}$ denote the actual sample value and the predicted sample value by the adaptive CNN structure, respectively. m denotes the number of samples; ρ_1 and ρ_2 denote weighted modulus.

3.4. CNN Architecture Based on Multifeature Fusion. The input of the above mentioned adaptive CNN structure is the residual blocks of adjacent blocks, but the edge CUs do not have enough reference blocks. In order to improve the accuracy of classification, we propose a CNN structure based on multifeature fusion. Furthermore, we evaluate the prediction accuracy based on the difference in the number of parameters to determine the most suitable CNN based on multifeature fusion. Table 2 shows the prediction accuracy of CNN based on multifeature fusion under different parameter setting (Conv, kernel, and FCL).

Figure 8 shows a flowchart of CNN architecture. Based on the traditional CNN structure, a CNN architecture based on the fusion of texture and depth feature is established, which uses the calculated SD and depth as the input of the CNN architecture and is divided into two channels. Each channel includes convolutional layers, pooling layers, and FCLs. The pooling layer is one of the effective tools in the CNN architecture, where the type of the pooling layer is the max pooling layer. In order to improve the performance of the CNN network, all convolutional layers and hidden FCLs are activated with ReLU. Moreover, the final output is classified by using the LF, which is used to improve the classification accuracy. With the above settings, the edge CUs in complex area are accurately classified.

Table 1: The prediction accuracy of adaptive CNN under different parameter setting.

(%)	Number of parameters			Number of inputs	
	Conv	Kernel	FCL	NB2 and NB4	NB1, NB2, NB3, and NB4
Default parameter	2	6	2	73.43	76.82
Comparison of Conv	3	6	2	72.88	74.94
	4	6	2	71.67	73.73
Comparison of kernel	2	8	2	80.55	82.76
	2	10	2	88.96	91.67
Comparison of FCL	2	6	3	70.59	73.80
	2	6	4	68.81	69.80

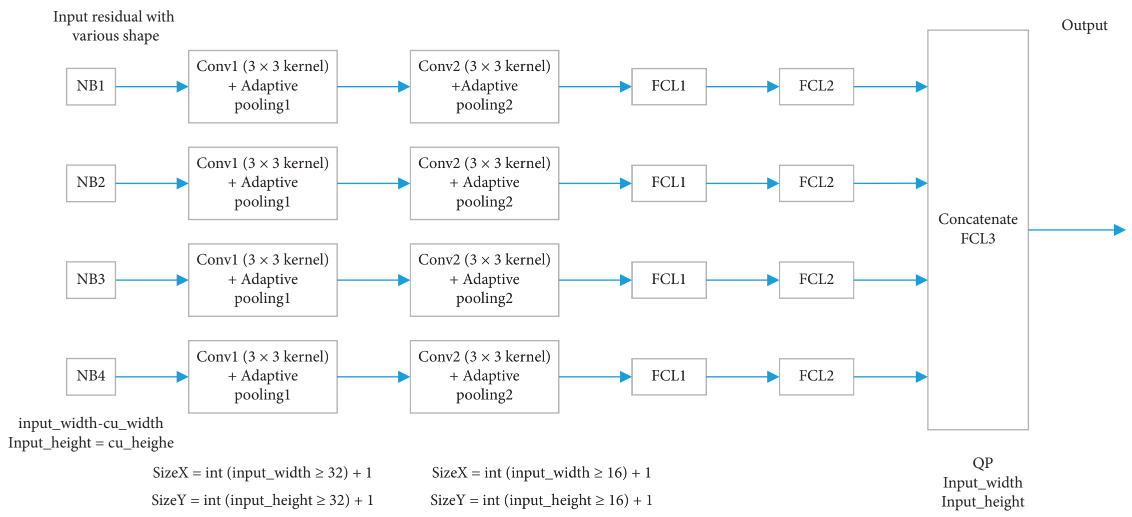

Input residual with various shape
Output
NB1
NB2
NB3
NB4
Conv1 (3 × 3 kernel) + Adaptive pooling1
Conv2 (3 × 3 kernel) +Adaptive pooling2
Conv2 (3 × 3 kernel) + Adaptive pooling2
FCL1
FCL2
Concatenate FCL3
input_width-cu_width
Input_height = cu_heighe
SizeX = int (input_width ≥ 32) + 1
SizeY = int (input_height ≥ 32) + 1
SizeX = int (input_width ≥ 16) + 1
SizeY = int (input_height ≥ 16) + 1
QP
Input_width
Input_height

Figure 7: The proposed adaptive CNN structure.

Table 2: The prediction accuracy of CNN based on multifeature fusion under different parameter setting.

(%)	Number of parameters			Accuracy
	Conv	Kernel	FCL	
Default parameter	4	6	2	71.85
Comparison of Conv	2	6	2	74.88
	3	6	2	79.67
Comparison of kernel	4	8	2	80.55
	4	10	2	89.96
Comparison of FCL	4	6	3	70.59
	4	6	4	73.81

3.5. CNNs Training. In the traditional CNN training scheme, all input feature sizes of the training samples are the same, the training data will be divided into several batches, and the training process is based on these batches. However, since the shapes of CUs in H.266/VVC are different, it is impossible to train CNNs with the previous methods. In the training process of the proposed adaptive CNN structure, the same size in the training data is classified into the same batch, and the different sizes are divided into different batches. Therefore, the proposed adaptive CNN architecture is trained in batches.

The size of all input sizes of the training samples is fixed in the traditional CNN training scheme. However, QTMT structure divides the CU into square and rectangle, which results in different shapes of the input CU. To solve this issue, different CU shapes are classified into different batches and the same shape CUs are grouped into a batch according to the size of the input training data in the training process of the proposed adaptive CNN method. We separately train these different shapes of CU. In training process, the training samples are firstly collected. The training samples involved in the training include the sequences "Tango2," "Catrobot1," "Kimono," "PartyScene," "RaceHorses," and "Johnny" under four QPs (22, 27, 32, and 37); there are 60 K in total. The videos with different resolutions make the training of CNN structure more accurate. Then, different sizes of CUs will be classified into different data sets. In addition, the input of the adaptive CNN structure is the residuals of the neighboring blocks of the current CU, which are separately trained in Figure 7. And the parameters of the adaptive CNN structure are not random, and the next training parameters are obtained from the previous training. Finally, the training of adaptive CNN structure is completed.

The CNN structure based on multifeature fusion uses the same training samples to the adaptive CNN structure during the training process. Then, the features are extracted from the training samples. Further, the extracted sample features are used as input of CNN structure based on the multifeature fusion, which are divided into two-way training as shown in

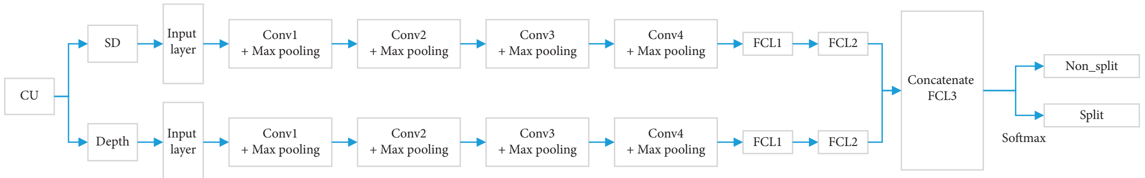


Figure 8: CNN architecture based on multifeature fusion.

Figure 8. Finally, we gain the trained CNN structure based on multifeature fusion.

4. Experimental Result

This paper incorporates heuristic algorithm and CNN methods. We compare SD value and threshold to effectively identify homogeneous areas. In order to further assess the accuracy of texture classification model based on threshold, P_e is defined as follows:

$$P_e = \frac{N_{\text{Split}}(h) + N_{\text{Non\_split}}(c)}{N_{\text{Split}} + N_{\text{Non\_split}}} \times 100\%, \tag{10}$$

where $N_{\text{Split}}(h)$ is the number of homogeneous CUs and $N_{\text{Non\_split}}(c)$ represents the number of complex CUs, which are misclassified CUs. Split denotes CU split and Non_split denotes CU nonsplit. Furthermore, N_{Split} and $N_{\text{Non\_split}}$ denote the number of split and nonsplit CUs, respectively. The texture classification model can achieve high classification accuracy by verifying, which can precisely classify textured areas, as shown in Table 3.

In order to evaluate the performance of the proposed algorithm, this method has been embedded in VTM7.0 to implement the simulation experiments. The test video sequences are encoded with default parameters in "All Intra" configuration. The BD-rate reflects the performance of the proposed method, and the time saving (TS) incarnates the reduction in complexity, which is defined as

$$\text{TS}(\%) = \frac{T_{\text{VTM7.0}} - T_{\text{proposed}}}{T_{\text{VTM7.0}}}, \tag{11}$$

where $T_{\text{VTM7.0}}$ denotes the encoding time of VTM7.0 and T_{proposed} denotes the encoding time of the proposed method.

Table 4 shows the coding performance of the proposed scheme. In the proposed method, the trained CNN models are embedded into the encoder to accelerate CU partition. In addition, the training time of CNN model is not counted in the total coding time. It is observed from Table 4 that the proposed method can decrease 39.39% coding time and increase the BD-rate to 0.86% compared with VTM7.0. Thus, the proposed method can effectively increase time savings and have a good RD performance.

In order to assess the performance of the proposed method, we compare the proposed method with state-of-the-art fast methods of H.266/VVC, which use the same configuration file to encode the test videos for fair comparison. The experimental results of the proposed method are compared with VTM7.0, while the experimental results of these methods in FPIC [18], ACSD [26], and FCPD [31] are compared with VTM4.0. It can be observed from Table 5 that the experimental results of the proposed method can save 39.31% coding time and maintain a similar RD performance compared with VTM7.0. We can see from Table 5 that FPIC, ACSD, and FCPD method have good RD performance, but the coding time savings of ACSD and FCPD method are less than the proposed algorithm. Among the four methods, the BD-rate of the proposed method is the smallest, while the BD-rate of the FPIC method is the largest. It is observed that FPIC, ACSD, and FCPD schemes have average BD-rate increase of 1.39%, 0.99%, and 1.37%, respectively. Compared with the proposed algorithm, the average BD-rate of FPIC, ACSD, and FCPD method increases by 0.52%, 0.12%, and 0.50%, respectively. The coding time saving of the proposed algorithm is less than FPIC method, but the average BD-rate increase is less than that of FPIC algorithm. Furthermore, the coding time savings of the proposed algorithm are better than ACSD and FCPD method. It can be seen from Table 5 that the time savings of ACSD and FCPD method increase by 5.82% and 9.78% compared with the proposed method. The experimental results may fluctuate for different test videos, because high-definition (HD) or ultra-high-definition (UHD) videos tend to have larger CUs. And more misclassification results can lead to more BD-rate. The proposed algorithm attains better coding property, which is primarily due to the threshold model of the heuristic method and two improved CNN structures.

Figure 9 shows RD performance for the proposed method compared with VTM7.0 in test videos. It is observed from Figure 9 that the proposed scheme can achieve consistent performance in terms of RD performance compared with VTM.

Figure 10 shows the simulation results of the proposed scheme and state-of-the-art fast methods including FPIC, ACSD, and FCPD method. Figures 10(a) and 10(b) show the results of time savings and BD-rate, respectively. Compared with ACSD and FCPD algorithm, the proposed scheme has higher performance in reducing computational burden, which can further save about 5.78%–9.82% encoding time. Compared with FPIC, ACSD, and FCPD algorithm, our proposed method has better coding efficiency, which can further reduce 0.12%–0.52% BD-rate. Such results show that the proposed scheme is effective for all classifications of test videos and the computational complexity is better than the state-of-the-art fast methods of H.266/VVC.

Table 3: The classification probability of the texture classification model.

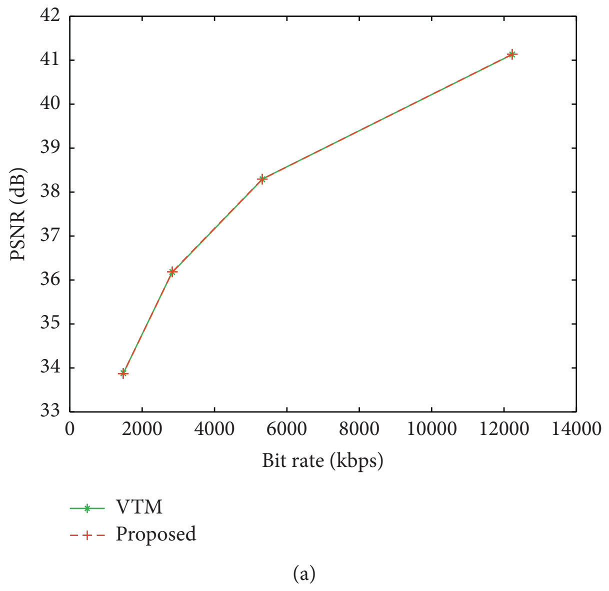
Test sequence		P_e			
Level (%)		Depth = 0	Depth = 1	Depth = 2	Depth = 3
Class B (1920 × 1080)	ParkScene	5.35	7.12	8.24	9.37
	Cactus	4.13	6.22	5.68	5.18
Class C (832 × 480)	BQMall	2.14	4.95	5.12	6.3
	PartyScene	—	4.21	4.62	4.99
Class D (416 × 240)	BasketballPass	—	6.76	5.81	4.75
	RaceHorses	—	5.11	6.15	7.04
	Average	3.87	5.73	5.94	6.37

Table 4: The coding performance of the proposed algorithm compared with VTM7.0.

Test sequence		The proposed method	
		BD-rate (%)	TS (%)
Class A1	Tango2	0.78	38.87
	FoodMarket4	0.82	39.96
	Campfire*	0.89	41.55
Class A2	Catrobot1	0.92	40.14
	DaylightRoad2	0.85	39.58
	ParkRunning3*	0.81	38.22
Class B	Kimono	0.78	37.51
	ParkScene	0.61	39.56
	BQTerrace	0.76	41.79
Class C	PartyScene	0.37	36.73
	RaceHorsesC	0.24	30.68
	BasketballDrill	1.25	39.21
Class D	BlowingBubbles	0.83	40.87
	RaceHorses	0.56	36.51
	BQSquare	0.58	36.67
Class E	Johnny	1.56	43.78
	FourPeople	1.34	46.51
	KristenAndSara	1.57	40.85
	Average	0.86	39.39

Table 5: The coding performance of the proposed algorithm compared with VTM7.0 and previous works.

Test sequence		The proposed		FPIC [18]		ACSD [26]		FCPD [31]	
		BD-rate (%)	TS (%)	BD-rate (%)	TS (%)	BD-rate (%)	TS (%)	BD-rate (%)	TS (%)
Class B 1920 × 1080	Kimono	0.78	37.51	1.72	66.59	0.87	33.32	1.98	41.82
	ParkScene	0.61	39.56	1.28.	56.28	0.83	35.41	1.38	31.60
	BQTerrace	0.76	41.79	1.16	49.44	0.95	34.50	1.19	29.47
Class C832 × 480	PartyScene	0.37	36.73	0.28	41.71	0.55	31.10	1.05	35.23
	RaceHorsesC	0.24	30.68	0.84	52.07	0.37	26.63	2.96	33.89
	BasketballDrill	1.25	39.21	1.91	53.05	1.30	33.39	1.36	28.73
Class D416 × 240	BlowingBubbles	0.83	40.87	0.49	43.90	0.95	33.90	0.73	21.87
	RaceHorses	0.56	36.51	0.54	44.93	0.71	31.79	1.59	31.83
	BQSquare	0.58	36.67	0.17	32.34	0.68	30.73	-0.11	23.00
Class E1280 × 720	Johnny	1.56	43.78	3.07	62.55	1.63	38.73	1.51	24.44
	FourPeople	1.34	46.51	2.55	62.18	1.38	38.01	1.37	26.65
	KristenAndSara	1.57	40.85	2.56	60.82	1.61	34.84	1.53	25.32
	Average	0.87	39.31	1.39	52.16	0.99	33.53	1.37	29.49



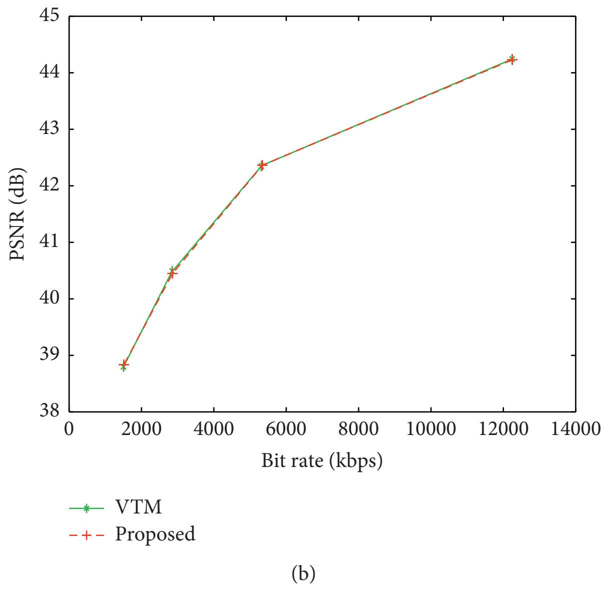
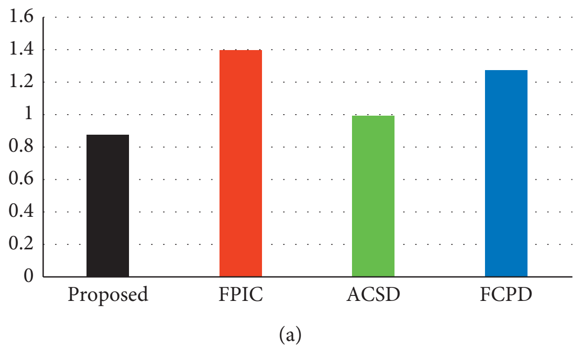


FIGURE 9: RD performance of the proposed method. (a) RD of “PartyScene.” (b) RD of “FourPeople.”



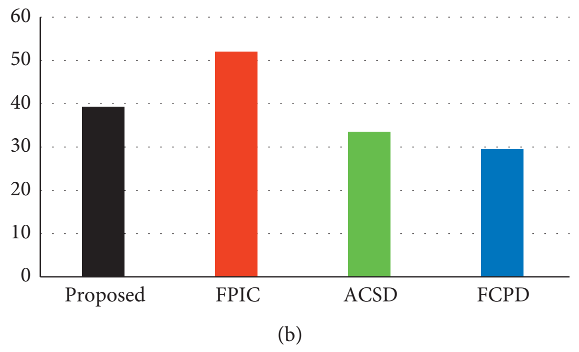


FIGURE 10: Results of the proposed method and previous methods. (a) BD-rate increase. (b) Time saving.

5. Conclusion

This paper proposes an adaptive CU split decision algorithm based on deep learning and multifeature fusion. This method develops a texture classification model based on threshold to recognize complex and homogeneous CUs. According to the adaptive CNN structure and CNN structure based on multifeature fusion, the CUs of various shapes are classified. The CU partition is determined by the trained network and various parameters of CU. The proposed method can successfully process the size of various training samples and avoid the RDO calculation. The experimental results show the good performance of the proposed algorithm to achieve fast encoding, which can save 39.39% of the encoding time. Furthermore, the proposed method is superior to the state-of-the-art methods with a better complexity reduction.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (nos.61771432, 61302118, and 61702464) and the Scientific and Technological Project of Henan Province (no. 202102210179).

References

[1] B. Bross, *Versatile Video Coding (Draft 1), JVET-J1001*, Joint Video Exploration Team (JVET), San Diego, CA, USA, 2018.
[2] The VVC test model 1, http://jvet.hhi.fraunhofer.de/svn/svn_WCSoftware_VTM/tags/VTM-1.0/.
[3] S. L. B. Brass and J. Chen, *Versatile Video Coding (Draft 2)*, vol. 22, pp. 1649–1668, 11th JVET Meeting, Ljubljana, Slovenia, Apr. 2018.

[4] M. Karczewicz and E. Alshina, *JVET AHG Report: Tool Evaluation (AHG1), JVET-F0001*, JVET 6th Meeting, Hobart, Australia, 2017.

[5] A. Said, X. Zhao, M. Karczewicz, J. Chen, and F. Zou, "Position dependent prediction combination for intra frame video coding," in *Proceedings of 2016 IEEE International Conference on Image Processing (ICIP)*, pp. 534–538, IEEE, Phoenix, AZ, USA, September 2016.

[6] X. Zhao, J. Chen, M. Karczewicz, L. Zhang, X. Li, and W. J. Chien, "Enhanced multiple transform for video coding," in *Proceedings of Data Compression Conference*, pp. 73–82, IEEE, Snowbird, UT, USA, March 2016.

[7] F. Bossen, X. Li, and K. Suehring, *AHG Report: Test Model Software development (AHG3), JVET-J0003*, Joint Video Exploration Team (JVET), San Diego, CA, USA, 2018.

[8] J. Chen, Y. Ye, and S. H. Kim, *Algorithm Description for Versatile Video Coding and Test Model 8 (VTM 8): Document JVET-Q2002*, Joint Video Experts Team (JVET) of ITU-T and ISO/IEC, Brussels, Belgium, 2020.

[9] H. Yang, L. Shen, X. Dong, Q. Ding, P. An, and G. Jiang, "Low complexity CTU partition structure decision and fast intra mode decision for versatile video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 6, pp. 1668–1682, 2019.

[10] Z. Wang, S. Wang, J. Zhang, and S. Ma, "Local-constrained quadtree plus binary tree block partition structure for enhanced video coding," in *Proceedings of 2016 Visual Communications and Image Processing (VCIP)*, pp. 1–4, IEEE, Chengdu, China, November 2016.

[11] Z. Jin, P. An, L. Shen, and C. Yang, "CNN oriented fast QTBT partition algorithm for JVET intra coding," in *Proceedings of 2017 IEEE Visual Communications and Image Processing (VCIP)*, pp. 1–4, IEEE, St. Petersburg, FL, USA, December 2017.

[12] T. Lin, H. Jiang, J. Huang, and P. Chang, "Fast binary tree partition decision in H.266/FVC intra Coding," in *Proceedings of 2018 IEEE International Conference On Consumer Electronics-Taiwan (ICCE-TW)*, IEEE, Taichung, Taiwan, pp. 1-2, May 2018.

[13] S. De-Luxán-Hernández, H. Schwarz, D. Marpe, and T. Wiegand, "Fast line-based intra prediction for video coding," in *Proceedings of 2018 IEEE International Symposium on Multimedia (ISM)*, pp. 135–138, IEEE, Taichung, Taiwan, December 2018.

[14] T. Fu, H. Zhang, F. Mu, and H. Chen, "Fast CU partitioning algorithm for H.266/VVC intra-frame coding," in *Proceedings of 2019 IEEE International Conference On Multimedia And Expo (ICME)*, pp. 55–60, IEEE, Shanghai, China, July 2019.

[15] J. Chen, Y. Chiu, C. Lee, and Y. Tsai, "Utilize neighboring LCU depth information to speedup FVC/H.266 intra coding," in *Proceedings of 2019 International Conference On System Science And Engineering (ICSSE)*, pp. 308–312, IEEE, Dong Hoi, Vietnam, July 2019.

[16] M. Lei, F. Luo, X. Zhang, S. Wang, and S. Ma, "Look-Ahead prediction based coding unit size pruning for VVC intra coding," in *Proceedings 2019 IEEE International Conference on Image Processing (ICIP)*, pp. 4120–4124, IEEE, Taipei, Taiwan, September 2019.

[17] S.-H. Park and J.-W. Kang, "Context-based ternary tree decision method in versatile video coding for fast intra coding," *IEEE Access*, vol. 7, pp. 172597–172605, 2019.

[18] J. Chen, H. Sun, J. Katto, X. Zeng, and Y. Fan, "Fast QTMT partition decision algorithm in VVC intra coding based on variance and gradient," in *Proceedings of 2019 IEEE Visual Communications and Image Processing (VCIP)*, pp. 1–4, IEEE, Sydney, Australia, December 2019.

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

[20] 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.

[21] T. Li, M. Xu, and X. Deng, "A deep convolutional neural network approach for complexity reduction on intra-mode HEVC," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1255–1260, IEEE, Hong Kong, China, July 2017.

[22] M. Xu, T. Li, Z. Wang, X. Deng, R. Yang, and Z. Guan, "Reducing complexity of HEVC: a deep learning approach," *IEEE Transactions on Image Processing*, vol. 27, no. 10, pp. 5044–5059, 2018.

[23] T. Amestoy and A. Thomas, "Random forest oriented fast QTBT frame partitioning," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1837–1841, IEEE, Brighton, UK, May 2019.

[24] Z. Jin, P. An, C. Yang, and L. Shen, "Fast QTBT partition algorithm for intra frame coding through convolutional neural network," *IEEE Access*, vol. 6, pp. 54660–54673, 2018.

[25] S. Ryu and J. Kang, "Machine learning-based fast angular prediction mode decision technique in video coding," *IEEE Transactions on Image Processing*, vol. 27, no. 11, pp. 5525–5538, 2018.

[26] G. Tang, M. Jing, X. Zeng, and Y. Fan, "Adaptive CU split decision with pooling-variable CNN for VVC intra encoding," in *Proceedings of IEEE Visual Communications and Image Processing (VCIP)*, pp. 1–4, IEEE, Sydney, Australia, December 2019.

[27] T. Amestoy, A. Mercat, W. Hamidouche, D. Menard, and C. Bergeron, "Tunable VVC frame partitioning based on lightweight machine learning," *IEEE Transactions on Image Processing*, vol. 29, no. 1, pp. 1313–1328, 2020.

[28] F. Galpin, F. Racapé, S. Jaiswal, P. Bordes, F. Le Léannec, and E. François, "CNN-based driving of block partitioning for intra slices encoding," in *Proceedings of 2019 data compression conference (DCC)*, IEEE, Snowbird, UT, USA, pp. 162–171, March.2019.

[29] Z. Wang, S. Wang, X. Zhang, S. Wang, and S. Ma, "Fast QTBT partitioning decision for inter frame coding with convolution neural network," in *Proceedings of 25th IEEE International Conference on Image Processing (ICIP)*, pp. 2550–2554, IEEE, Athens, Greece, October 2018.

[30] Y. Zhang, G. Wang, R. Tian, M. Xu, and C. C. J. Kuo, "Texture-classification accelerated CNN scheme for fast intra CU partition in HEVC," in *Proceedings of 2019 Data Compression Conference. (DCC)*, pp. 241–249, IEEE, Snowbird, UT, USA, March 2019.

[31] N. Tang, J. Cao, F. Liang et al., "Fast CTU partition decision algorithm for VVC intra and inter coding," in *Proceedings of IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, pp. 361–364, IEEE, Bangkok, Thailand, November 2019.