IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 17 March 2023, accepted 3 April 2023, date of publication 10 April 2023, date of current version 13 April 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3266002

RESEARCH ARTICLE

Fast CU Decision Algorithm Based on Texture Complexity and CNN for VVC

HONGCHAN LI, PENG ZHANG, BAOHUA JIN, AND QIUWEN ZHANG
College of Computer and Communication Engineering, Zhengzhou University of Light Industry, Zhengzhou 450002, China

Corresponding author: Peng Zhang (332107010564@e-mail.zzuli.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61771432 and Grant 61302118; in part by the Basic Research Projects of Education Department of Henan under Grant 21zx003 and Grant 20A880004; in part by the Key projects Natural Science Foundation of Henan under Grant 232300421150; in part by the Scientific and Technological Project of Henan Province under Grant 232102211014; and in part by the Postgraduate Education Reform and Quality Improvement Project of Henan Province under Grant YJS2021KC12, Grant YJS2023JC08, and Grant YJS2022AL034.

ABSTRACT Versatile Video Coding (VVC) introduces many advanced video coding techniques. These advanced video coding techniques not only improve video compression efficiency and video quality, but also greatly increase coding time and computational complexity. Therefore, this paper uses a fast CU partitioning decision algorithm based on texture complexity and convolutional neural networks (CNN). First, we performed statistics and analysis of CU segmentation patterns for videos in the standard training set, then processed large blocks of CU based on texture complexity. To realize the purpose of fast partitioning, we design different CNN models for CU with different segmentation modes. In the design of CNN model, we use the symmetric convolutional kernel and asymmetric convolutional kernel to extract features in different directions effectively. In the loss function, we used the cross-entropy function to train the CNN model to improve the accuracy of the model. Finally, a double threshold is set in the candidate list to achieve a compromise between coding performance and coding complexity. Experimental results show that, compared to the VTM10.0 anchoring algorithm, our fast scheme, in terms of encoding time, decreases by 55.90% and BDBR increases by 1.79%; our moderate scheme, in terms of encoding time, decreases by 47.90%. BDBR increases by only 1.29%.

INDEX TERMS CNN, asymmetric convolutional kernel, symmetric convolutional kernel, VVC.

I. INTRODUCTION

With the pursuit of high-resolution video, full high definition (FHD) video is no longer enough to meet people's needs, and 4K and 8K resolution videos have come into being. The ensuing network transmission pressure is huge. In this case, High Efficiency Video Coding (HEVC) is no longer effective in compressing large amounts of data. Therefore, the Joint Video Experts Group (JVET) of the Video Coding Experts Group (VCEG) of the International Telecommunication Union - Telecommunication Standardization Sector (ITU-T) and the Motion Picture Experts Group (MPEG) of the International Organization for Standardization/ International Electrotechnical Commission (ISO/IEC) worked on the development of the next generation video coding standard, namely the Versatile Video Coding (VVC) [1].

HEVC is based on the traditional macroblock encoding method, which uses coding tree units. In efficient video coding, each frame of the picture is divided into coding units (CTUs), which can be further divided into coding units (CUs) by quadtree division, and CUs can also be divided into four smaller sub-CUs by quadtree division. Both CTU and CU are square. the maximum size of CTU is 64×64, the minimum size is 16×16. the maximum size of CU is 64×64, the minimum size is 8×8. Unlike HEVC, VVC uses a new block division structure with a nested multi-type of tree (QTMT). It is due to the QTMT division that CU can be divided not only into squares but also rectangles to better accommodate video content with different texture characteristics. As a result, the encoding efficiency is greatly improved. Compared with

The associate editor coordinating the review of this manuscript and approving it for publication was Zhaoqing Pan.

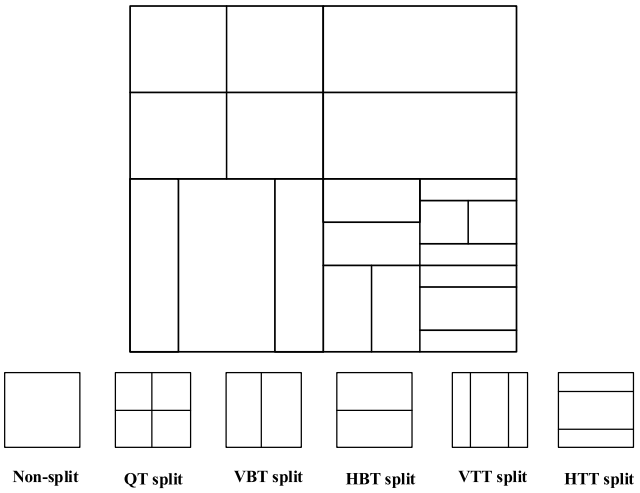

FIGURE 1. Partition structure and six split modes.

HEVC, the QTMT division used in VVC can generate more types of subunits and can better adapt to different video contents. In VVC, there are six ways to classify CU, as shown in Figure 1: Non-split (NS), quadrinomial tree (QT), vertical binary tree (BTV), horizontal binary tree (BTH), vertical trinomial tree (TTV), and horizontal trinomial tree (TTH) [2]. These six division patterns can form a candidate list, and the optimal pattern n^* is the pattern with the smallest RD Cost in the candidate list n. The two calculation equations are as follows:

$$RD\ Cost = SSE + \lambda \times Bit_{mode} \tag{1}$$

$$n^* = n_{min_{RD\ Cost}} \tag{2}$$

where SSE is the distortion of luminance and chrominance, Bit_{mode} is the cost of in-frame prediction, and λ is the Lagrangian multiplier. n_1 contains six division modes, NS, QT, VBT, HBT, VTT, and HTT, and n_2 contains five division modes, NS, VBT, HBT, VTT, and HTT.

Since VVC adopts a new block partitioning structure of nested multitype trees (QTMT), the segmentation structure is optimized to better match the texture features of the image, the video quality is significantly improved, and the prediction residuals are reduced [2]. Compared to HEVC [3], VVC not only provides improvements in video quality and reduces the time used in the encoding process. Therefore, for the new block partitioning structure of nested multi-type trees (QTMT), we used a combination of traditional methods and convolutional neural networks.

We used a gradient-based method and a neural network-based method to realize the prediction of CU division patterns. First, we now use an algorithm based on texture complexity to determine whether CU is divided for CU with large size, and then, we use a fast CNN-based CU division algorithm for CU with multiple division patterns to accommodate fast CU division decisions for VVC. Then explain the CNN model in detail, and finally select the optimal segmentation mode by adjusting the threshold in the candidate list. Our method guarantees both the reduction of the time of the encoding process and the loss of video quality within an acceptable range.

Section II reviews the related work of previous generations in reducing the computational complexity of VVC. Section III presents a statistical kernel analysis on a standard dataset. Section IV presents our pre-decision algorithm based on neural networks using texture complexity-based algorithms. In Section V, we detail the results of the experiments with the standard training video set and an analysis of the results, and Section VI summarizes the algorithms used in this paper and the expected results achieved.

II. RELATED WORK

To achieve to computational complexity reduction in video coding. In [4], Otsu's method is used by measuring the complexity of the LCU texture and deciding whether to skip the CU depth level or not. Since the predicted patterns are related to the gradient direction, the Sobel operator is improved to achieve the purpose of measuring the gradient direction of the PU to reduce the in-frame patterns in the candidate list. In [5], two algorithms are used, in the VVC coding process, by combining a multi-pair Bayesian algorithm and an improved coding unit division decision algorithm with a de-block filter, which can discard some unnecessary candidate patterns in advance to achieve a reduced complexity of video coding. In [6], a fast decision algorithm based on variance and gradient is used with the aim of solving the asymmetric partitioning problem in VVC by minimizing the time used in the encoding process without losing video quality. A method incorporating a fast texture energy-based CU partitioning decision algorithm and a texture orientation-based CU segmentation pattern decision algorithm is proposed in [7]. The method uses texture features to predict the segmentation mode of coding units and discards unnecessary candidate patterns to achieve the goal of reducing the computational complexity and computational effort of intra-frame coding in VVC. In [8] the horizontal and vertical texture complexity is evaluated based on the sum of the mean absolute deviation (SMAD) of the sub-blocks, and then the ratio of the SMAD of the threshold and texture direction is used to predict the best segmentation pattern. In the literature [9], a novel method of pattern decision making by inserting a decision tree is proposed, which significantly reduces the encoding process used is the traitor. In [10], it is proposed to divide CU blocks into horizontal and vertical division methods based on texture direction in advance based on gradient, skipping unnecessary division patterns in advance. In the literature [11], 4K video is first analysed and modelled, using a decision tree approach that takes full account of its features to improve the accuracy of the model predictions.

Deep learning provides an effective solution to the classification problem through deep convolutional neural networks (CNN) when it comes to CU segmentation decisions. In the literature [12], the texture and gradient information of

the CU is analysed to select the optimal division pattern to end the division early. In the literature [13], a unique CNN model is proposed which has the function of early exit from coding unit division. The early exit mechanism is used to speed up the coding process. In the literature [14], an intra-frame pattern decision algorithm based on the random forest classifier model is designed to optimize the algorithm for intra-frame pattern prediction. In the literature [15], the probability of dividing a size 64 × 64CU into 4 × 4 is calculated for its boundary vector, and then the probability of dividing a size 64 × 64CU downwards is deduced in reverse. In literature [16], an early termination is hierarchical CNN is proposed instead of the process of searching RD Cost using brute force. In the literature [17], coding units of different sizes are classified and then CNN models are designed for their unique features to improve the coding efficiency. In literature [18], a fast ResNet-based CU partitioning decision algorithm is proposed to predict CU segmentation patterns by CNN models to shorten the coding time. In [19], the decision of whether to terminate the division early is made by judging the extracted features and deciding whether to skip the horizontal or vertical partition early. In the literature [6], texture complexity is used to detect whether CU is partitioned or not. In the literature [20], the extraction of gradient features with the Sobel operator was proposed to solve the non-square classification problem in the coding process. In the literature [21], the coding time is shortened by improving the ETH-CNN model by terminating the block division earlier. In the literature [22], a deep learning approach to drive CTU partitioning is used instead of traditional algorithms to speed up the encoding. In the literature [15], the difficulty of block partitioning is reduced by predicting the probability vector through neural networks.

III. STATISTICS AND ANALYSIS

The goal of this section is to statistically and analyse the classification patterns of different CUs. First, the video sequence we used was the JVET test set, and we selected three standard video sequences with different resolutions. Video sequences include Johnny, Campfire, Basketball Drive and Basketball Pass. We not only coded them, but also counted the proportion of CU segmentation patterns. Detailed statistics and analysis are performed through video sequences of different resolutions and sequences of different video contents of the same resolution.

We counted 4716524 CUs with CU size ranging from 32 × 32 to 16 × 16. From Table 1, we can visualize the proportion of different CU segmentation patterns in the four video sequences.

1) Among the CUs with different sizes, we can see that the lowest percentage without division is 17.43% and the highest percentage is 42.38%. Therefore, early termination of CU division not only reduces the complexity in the coding process, but also saves the time used in the coding process.
2) The size of 32 × 32CU, HTT and VTT have the lowest percentage among the six segmentation modes, with an average of 8.27% and 7.34%, respectively. Therefore, we focus our analysis on the other four segmentation modes. Thus, CNN models are constructed for CU of size 32 × 16 and 16 × 32.

IV. PROPOSED METHOD

A. THE OVERALL ALGORITHM

Unlike HEVC, VVC adopts the QTMT division, and VVC contains 6 division modes. In VVC intra-frame CU division, the division mode of each CU is determined using a brute force search of the RD Cost. The search approach consists of two parts: a top-down process of checking the RD Cost of the child CUs and a bottom-up process of comparing the RDC of the child CUs with the RDC of the parent CUs. In this search approach, there will be many redundant computations, therefore, we propose an early termination of CU division and reduction of redundant coding computations by predicting the CU division pattern. Compared with traditional methods, CNN can effectively solve the classification and prediction problems. Figure 2 shows the flowchart of the algorithm, which firstly performs gradient checking for size 64 × 64CU to determine whether to divide, if not, it directly exits the encoding, and if the division condition is satisfied, then the division is performed. Then the sub-CU is predicted by the CNN model for the division pattern, and the optimal division pattern is selected by Equation 8. If the optimal division pattern is NS, the encoding is ended, and if the optimal pattern is not NS, the encoding continues to return to the CNN model until the CU size and depth do not satisfy the judgment condition, and the encoding is ended.

B. PRE-DECISION ALGORITHM WITH GRADIENT-BASED TEXTURE COMPLEXITY

The purpose of the pre-decision method is to decide whether to segment the CU by judging the texture complexity. Processing and analysis of the trained dataset, a size of 64 x 64CU can only produce two division patterns. Therefore, we will not use the neural network model to process them to avoid wasting unnecessary time. We propose a gradient-based approach to detect texture complexity and determine whether CU is divided. the gradient of CU is calculated by the Sobel operator with the following equation.

$$grad = \frac{\sum_{i=1}^{w}\sum_{j=1}^{h}\left(g_x^2 + g_y^2\right)}{w \times h} \tag{3}$$

where, g_x and g_y are the gradients in the x and y directions computed by the Sobel operator. Wand H denote the width and height of the CU. After calculating the gradient, we can need to determine whether the CU is divided according to the size of the gradient by comparing the gradient with the square of the quantization parameter, where the lower limit can be found in the article [23] and the specific judgment condition

TABLE 1. The proportion of different divisions in CUs of different sizes.

CU Size	Split mode	Johnny (525665)	Campfire (2597158)	Basketball Drive (1546520)	Basketball Pass (47179)	Average (1179131)
32×32	NS	20.17	24.70	15.04	9.80	17.43
	QT	26.61	19.17	21.58	41.04	27.10
	HBT	14.34	16.76	30.81	20.60	20.63
	VBT	23.48	25.69	15.34	12.06	19.14
	HTT	5.53	4.97	11.56	11.02	8.27
	VTT	9.78	8.71	5.37	5.48	7.34
32×16	NS	43.80	44.61	40.26	33.54	40.56
	HBT	14.23	14.57	28.36	26.39	20.89
	VBT	21.32	23.45	12.29	14.71	17.94
	HTT	6.80	4.17	14.12	17.35	10.61
	VTT	13.85	13.20	4.97	8.02	10.01
16×32	NS	40.07	50.86	42.56	36.04	42.38
	HBT	16.67	17.24	17.55	20.59	18.01
	VBT	23.57	17.63	20.32	18.44	19.99
	HTT	7.97	8.04	11.14	15.34	10.62
	VTT	11.66	6.05	8.43	9.60	8.935
16×16	NS	31.30	39.35	29.12	17.40	29.29
	QT	7.16	5.38	13.98	14.99	10.38
	HBT	17.81	19.12	24.75	25.31	21.75
	VBT	26.05	20.77	17.94	22.73	21.87
	HTT	6.47	7.40	9.03	11.23	8.53
	VTT	11.22	7.99	5.18	8.34	8.18

is known by Equation 4.

$$Result = \begin{cases} Non_{split} & grad < 0.15 \times QP^2 \\ split & (other) \end{cases} \quad (4)$$

When the gradient is less than $0.15\times QP^2$, the size is 64×64CU without division, and when the gradient is greater than $0.15\times QP^2$, the size is 64×64CU for quadtree division. Compared to neural network models, pre-decision algorithms do not require a network model to be built or a model to be trained, saving time in training the model compared to CNNs.

C. STRUCTURE OF CNN

Different from HEVC, VVC adopts a new block division structure of nested multitype tree (QTMT). Based on the asymmetry of MT division, we improve the CNN model to improve the model's ability to extract asymmetric features. The model is used to predict the probability of dividing patterns, which can effectively reduce the coding complexity. First, we obtain the CU of the size we need to input into the CNN, extract the features of the CU through different convolutional layers, then stitch the extracted features into a feature vector, and then perform feature fitting through different connection layers, and finally output a 1×n prediction probability vector, where the size of n is equal to the number of CU division patterns, which correspond to different patterns in the candidate list. In addition, since QP also has an impact on the division mode of CU. Therefore, we will add QP as an external feature to the fully connected layer. Then the activation function we use is SoftMax. In this section, we will introduce our neural network model and its important structure.

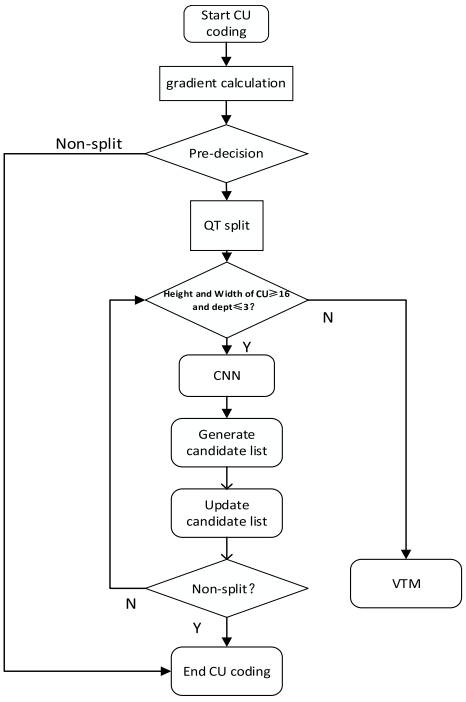

FIGURE 2. The algorithm flow chart.

V. CONVOLUTION LAYER

The convolutional layers designed for different CU sizes are different. There are six division modes for CU of size 32×32 and 16×16: no division, quadtree, vertical binary tree, horizontal binary tree, vertical trinomial tree, and horizontal trinomial tree. Therefore, we replace the symmetric

convolution kernels with asymmetric convolution kernels to be able to extract the features in different directions effectively. 32 × 32 CNN model is shown in Figure 3(a), and the structure contains three convolutional layers with three branches in each layer. The size of the branch convolution kernels in the first layer is 2 × 4, 4 × 4, and 4 × 2 (the number of filters is 16), and the size of the convolution kernels in each branch of the second and third layers is 1 × 2, 2 × 2, and 2 × 1 (the number of filters is 24 and 36), respectively. The 32 × 16 CNN model is shown in Figure 3(b), and the structure contains two convolutional layers with three branches in each layer. The size of the convolutional kernels in the first branch of the layer is 2 × 4 and 4 × 2 (the number of filters is 16), and the size of the convolutional kernels in the second and third layers is the same, and the convolutional kernels in each branch are 1 × 2, 2 × 2, and 2 × 1 (the number of filters is 24 and 36, respectively). While 32 × 16 and 16 × 32 have no QT division and only five division patterns, so we use non-pairwise convolutional kernels to extract features in different directions. 16 × 16 CNN model is shown in Figure 3(c), and the structure contains three convolutional layers with three branches in each layer. The size of the convolutional kernels of the first branch is 2 × 4, 4 × 4, and 4 × 2 (the number of filters is 16), and the size of the convolutional kernels of the second and third layers are the same, and the convolutional kernels of each branch are 1 × 2, 2 × 2, and 2 × 1 (the number of filters is 24 and 36), respectively.

VI. FULLY CONNECTED LAYER

The features extracted in the feature extraction stage are stitched together and spliced into 1 × 2176, 1 × 1024, and 1 × 544 feature vectors. Feature vectors of size 32 × 32 CU are then feature-fitted by fully connected layers of 2176 × 128, 128 × 96, and 96 × 6. Feature vectors of size 32 × 16 CU are then feature-fitted by fully connected layers of 1024 × 256, 256 × 128, and 128 × 5. The feature vectors of CU with size 16 × 16 are then feature-fitted by fully connected layers of 544 × 64, 64 × 48, and 48 × 6. Again, because QP has a significant effect on the division of CU, QP is fused into the feature vectors of the fully connected layer as external features.

VII. LOSS FUNCTION

Cross-entropy can be used to measure the probability between two probability distributions. CU classification is a multi-classification problem, and there is an error between the predicted value and the true value generated by the CNN model, and the greater the difference between the two probability distributions, the greater the cross-entropy. Where p is the prediction vector and q are the label vector, the loss function is calculated as follows:

$$H(p,q) = -\sum_{i=1}^{n} p(x_i)\, log\,(q(x_i)) \tag{5}$$

$$Loss = H(p,q) + H(1-p, 1-q) \tag{6}$$

A. MODEL SELECTION

This section focuses on the pattern selection process, the main purpose of which is to balance the time and coding complexity used in the coding process. In VVC intraframe coding, CU of size 32 × 32 and 16 × 16 will generate six division patterns, while CU of size 16 × 32 will generate five patterns with probability after CNN. Then, the division pattern with higher probability is selected from the candidate list. Therefore, we need to introduce a threshold φ to train our CNN model. By using an appropriate threshold φ, the accuracy of our model in predicting split patterns is guaranteed while allowing as few split patterns as possible to make it into the final candidate list.

VIII. UPDATE CANDIDATE LIST

The CU passes through the CNN model will generate the probabilities of the corresponding patterns that make up our candidate list. To improve the accuracy of the model, we need to update the candidate list. Therefore, the candidate list is first sorted, and then the optimal division pattern in the candidate list is decided by threshold.

For size 32 × 32 and 16 × 16CU, there are six division patterns in the candidate list, while for size 32 × 16CU, there are five division patterns in the candidate list, so we introduce a threshold φ to select the appropriate division patterns in n'. The predicted probability vector P obtained from the CNN model is called first, and then the candidate list is sorted in descending order according to the size of the predicted probability vector P. The formula is as follows:

$$(L_1, P') = sort\,(L, P) \tag{7}$$

where L_1 is the candidate list after descending order and P' is the predicted probability vector after descending order; after sorting the candidate list, we need to set a threshold φ to select n' suitable division patterns and update the candidate list L' according to the obtained n. where $L' = \{L_j | 1 \leq j \leq n\}$, L_j is denoted as the J element in L. The updated candidate list is obtained by this method, and RD Cost comparison is performed for n($n_1 \leq n \leq n_2$) patterns in the updated candidate list.

IX. THRESHOLD SETTING

Given the input to determine the size of the CU, the prediction probability is obtained by the CNN model, and n division patterns are selected to predict how the CU will be divided. Specifically, when the n division modes of the CU of the determined size satisfy $\sum_{i=1}^{x} P'_i \geq \varphi, x \in \{1,2,3,4,5,6\}$, the candidate list is updated to compare the RD cost with the parent CU to determine the final division mode, thus avoiding the process of comparing all redundant RD costs and significantly reducing the coding time and coding complexity.

After passing through the CNN model, the sum of the vector probabilities of the resulting division pattern is 1. When the probability of one vector is much greater than the

FIGURE 3. Architecture of CNN models. (a) Architecture of CNN models for 32 × 32 CU. (b) Architecture of CNN models for 16 × 32 CU. (c) Architecture of CNN models for 16 × 16 CU.

probability of other vectors, the candidate list discards the other division patterns, leaving only the division patterns. This case minimizes the coding complexity and coding time. The accuracy of the model in predicting division patterns may be reduced when the probability of having two or more division patterns is close and relatively large. So, we test different threshold judgment schemes to meet different needs. In equation 8, We set the threshold value as $\varphi \in (0, 1)$. In the JVET test set, we use seven official standard video sequences to train our CNN model and set different thresholds to test the accuracy of our CNN model under the standard video sequence.

$$n = arg\ min\ x\ s.t. \sum_{k=1}^{x} P_i' \geq \varphi, \quad x \in \{1, 2, 3, 4, 5, 6\} \quad (8)$$

In the JVET test set, we used video sequences of different resolutions to train our models. By changing the threshold, the prediction accuracy of our model is improved. When the candidate list only outputs the pattern with the highest probability by default, the CNN model of size 32 × 32 CU has poor prediction accuracy, only about 72%. the prediction accuracy of CNN models of size 16 × 32, 32 × 16CU is about the same, and the model accuracy is slightly higher than that of 32 × 32 CNN model. Therefore, we need to adjust the parameter φ according to different requirements. when the parameter φ is larger, the more patterns in the candidate list, the more accurate the model prediction, the higher the video quality, and the longer the coding time.

For this purpose, we designed different threshold schemes to meet different requirements. Inspired by [24] at the time of testing. Through experiments, we obtain different experimental results, as shown in Figure 4. The size 32 × 32, 16 × 32 and 16 × 16CU are highly affected by the threshold value. In Table 2, we have designed two scenarios to meet the different requirements. The purpose of the express solution is to speed up the encoding process and reduce unnecessary calculations. The purpose of the modest scheme is to save more time in the encoding process while ensuring that the loss of video quality is within an acceptable range. Moderate encoding schemes have a larger threshold, better encoding

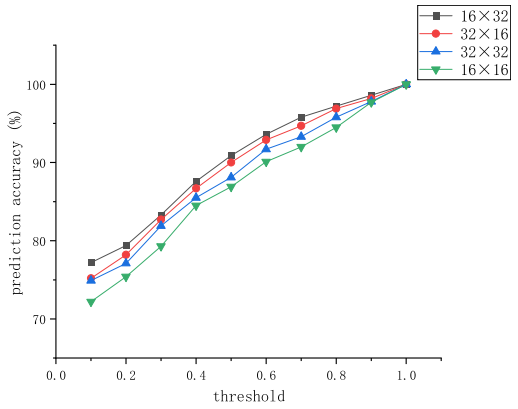

FIGURE 4. Relationship between the threshold φ and the accuracy of CNN model prediction.

TABLE 2. Threshold values for the model under both scenarios.

Scheme	φ	
	32×32、16×16	32×16、16×32
Fast	0.5	0.4
Moderate	0.6	0.5

performance, higher video quality, and more time savings in the encoding process than fast encoding schemes. Therefore, we can choose different schemes according to different requirements to achieve a compromise between reducing redundant computations in the encoding process and improving video encoding performance.

X. EXPERIMENTAL RESULTS

Firstly, the experimental equipment we used is described. Second, we perform experiments on the dual-threshold scheme and analyse the experimental results. Finally, the experimental results are compared and analysed with those of other algorithms to evaluate the proposed algorithm to reduce the time used in the encoding process while still maintaining a high-quality video picture. To demonstrate the feasibility of our algorithm.

A. EXPERIMENTAL CONFIGURATION

The algorithm proposed in this paper is implemented in VTM10.0, and both coding and CNN models are run on a computer equipped with an 12th Gen Intel(R) Core (TM) i9-12900H 2.50 GHz, 16GRAM and Windows OS. The video sequences of the test set can be divided into six types based on resolution. The resolutions of the six video sequences are shown in Table 3. The video sequences of the test set were encoded according to four quantization parameters QP (22, 27, 32, 37).

Compared to the BDBR and coding time savings (TS) of VTM10.0 to evaluate the performance of our model, the TS is calculated as follows:

$$TS = \frac{1}{4} \sum_{QP_i \in \{22,27,32,37\}} \frac{T_{VTM10.0}(QP_i) - T_{SC}(QP_i)}{T_{VTM10.0}(QP_i)} \times 100\% \quad (9)$$

TABLE 3. Resolution of six video sequences.

Class	Resolution
A1	3480×2160
A2	3480×2160
B	1290×1080
C	823×480
D	416×240
E	1280×720

where $T_{VTM10.0}$ is the anchor coding time of $T_{VTM10.0}$ and T_{SC} is the coding time of our proposed algorithm. In VVC, a common metric used in evaluating an algorithm is TS. The larger the TS, the shorter the coding time and the better the algorithm. From Equation (9), the shorter the encoding time of our proposed algorithm, the larger the TS, the superior the performance. Therefore, the larger the TS, the greater the performance improvement of the algorithm.

B. ANALYSIS OF EXPERIMENTAL PERFORMANCE

We tested 22 standard video sequences using two thresholding schemes to test the time of our algorithm during the encoding process and the performance associated with the encoding process. The statistical experimental results are shown in Table 4, from which under the fast scheme, the proposed algorithm saves 55.93% of the coding time and BDBR increases by 1.81%. Under the moderate scheme, the proposed algorithm saves 47.9% of the coding time and the BDRD increases by only 1.29%. Analysis of the experimental results shows that the fast scheme reduces the encoding time, but the BDRD increases too much. When comparing the high-resolution video, it can be found that the encoding time of the moderate scheme is only 7.97% more than that of the fast scheme, while the BDRD is reduced by 0.52%. The main reason is that the high-resolution video tends to be divided into smaller CUs when encoding, which leads to the growth of encoding time and thus video performance improvement, which is consistent with our experimental results. At low resolution video, the modest scheme increases the encoding time by only 7.59% compared to the fast scheme, while BDRD decreases by 0.47%. The main reason is that low-resolution videos tend to divide into larger CUs more, which reduces the coding complexity and coding time. In addition, the coding time of different video sequences fluctuates under the same resolution, for example: In the fast-encoding scheme, A1 takes 6.12% less time than the encoding process for A2 video sequences, while in the moderate encoding scheme, A1 takes 1.16% less time than the encoding process for A2 video sequences. So, the performance of the two schemes is about the same. In summary, the performance of the moderate coding scheme is significantly improved, and the algorithm can significantly reduce the time used for the coding process and reduce the coding complexity, resulting in better performance. Compare our proposed algorithm with previous Compare our proposed algorithm with previous algorithms,

TABLE 4. Comparison results of the dual-threshold scheme with the VTM10.0 encode.

Class	Sequence	Fast		Moderate	
		BDBR (%)	TS (%)	BDBR (%)	TS (%)
A1	Campfire	1.99	61.23	1.59	51.85
	FoodMarket4	2.38	60.95	1.61	50.11
	Tango2	2.25	58.63	1.55	50.59
A2	CatRobot1	2.35	58.99	1.77	47.92
	DaylightRoad2	1.96	61.32	1.39	54.33
	ParkRunning3	1.71	57.02	1.46	48.11
B	CaCTUs	1.86	55.11	1.31	44.95
	BQTerrace	1.62	56.74	0.94	48.33
	BasketballDrive	2.11	52.38	1.49	46.16
	Kimono	1.56	55.02	0.98	51.04
	PartyScene	1.43	56.55	1.05	51.18
C	RaceHorses	1.36	52.85	0.81	46.95
	PartyScene	1.02	53.51	0.83	45.88
	BQMall	1.48	54.64	1.24	48.33
D	BasketballDrill	1.82	52.27	1.18	45.17
	BasketballPass	1.93	50.08	1.41	40.04
	BlowingBubbles	1.39	53.16	0.99	43.86
	BQSquare	1.29	53.35	0.89	46.68
	RaceHorses	1.86	52.63	1.27	39.21
E	KristenAndSara	2.01	55.66	1.63	49.82
	FourPeople	1.73	58.39	1.55	52.64
	Johnny	2.37	60.07	1.58	50.67
	Average	1.81	55.93	1.29	47.90

TABLE 5. Compare the performance of our algorithm with the performance of others' algorithms.

Class	Sequence	Tang[19]		Fan[6]		Chen[20]		Propose	
		BDBR (%)	TS (%)	BDBR (%)	TS (%)	BDBR (%)	TS (%)	BDBR (%)	TS (%)
B	Kimono	0.52	35.05	1.93	59.51	1.72	66.59	1.07	50.34
	BQTerrace	0.81	35.29	1.08	45.30	1.16	49.44	0.89	48.02
	PartyScene	0.63	43.29	1.26	51.84	1.28	56.28	1.22	51.16
C	RaceHorses	0.65	30.53	0.88	49.05	084	52.07	0.88	47.69
	PartyScene	0.34	33.18	0.26	38.62	0.28	41.71	0.93	46.87
	BasketballDrill	1.30	29.62	1.82	48.48	1.91	53.05	1.37	44.06
D	RaceHorses	0.30	26.21	0.54	41.69	0.54	44.93	1.29	41.03
	BlowingBubbles	0.23	27.71	0.47	40.35	0.49	43.90	0.97	44.86
	BQSquare	0.22	26.02	0.19	31.95	0.17	32.34	0.93	46.82
E	KristenAndSara	0.94	47.20	3.22	56.88	2.56	60.82	1.59	49.65
	FourPeople	1.18	42.29	2.70	57.57	2.55	62.18	1.49	48.87
	Johnny	1.29	40.72	2.78	55.11	3.07	62.55	1.61	50.85
	Average	0.70	34.76	1.43	48.03	1.39	52.16	1.19	47.50

it can be seen from Table 5 that the two metrics we use are TS and BDRD.

In Table 5, comparing with BDRD, our algorithm significantly outperforms Chen and Fan in terms of BDRD. the experiments of Lin, Chen, and Fan were all conducted on VTM5.0, while our experiments were conducted on VTM10.0, which uses the same CU segmentation scheme as VTM5.0 and VTM10.0. Therefore, our comparison is reasonable. Overall, our algorithm has only 1.29% BDRD without wasting much coding time, so our algorithm is moderate. In encoding standard video sequences at different resolutions, we can see that the new way of encoding structure adopted by VVC is able to adapt to different video sequences. Our algorithm is more adaptable and significantly faster than the traditional VVC algorithm, saving more coding time.

Further RD performance analysis is performed for the algorithms we used. We compare our algorithm with the VTM10.0 anchoring algorithm under two video sequences, as shown in Figure 5, the video sequences are "FourPeople" and "Johnny", respectively. The video sequences are "FourPeople" and "Johnny" respectively. The horizontal coordinates indicate the bit rate, and the vertical coordinates indicate the Y-PSNR. At low QP, the difference between the RD curves of our proposed algorithm and the VTM10.0 anchoring algorithm is minimal and within an acceptable range. The main reason for this deviation is that a high-quality video picture is maintained while

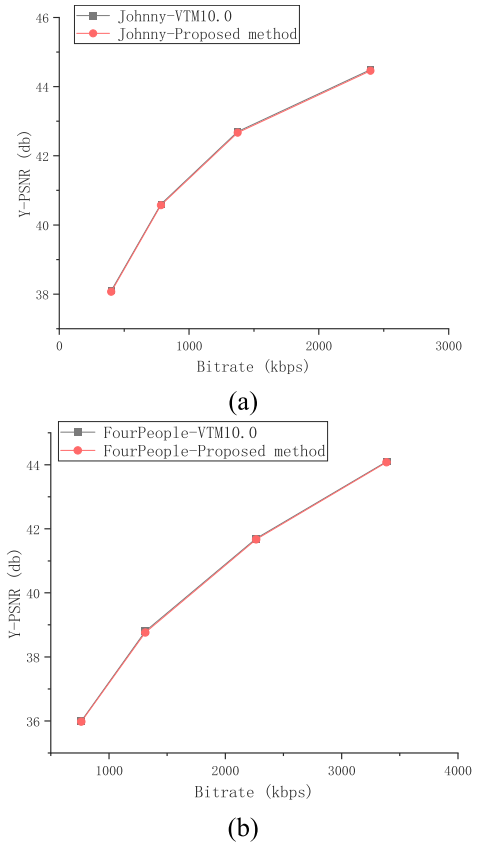

FIGURE 5. RD curves of "Johnny" and "FourPeople". (a) RD curves of "Johnny". (b) RD curves of "FourPeople".

ensuring a reduction of the time used in the encoding process.

XI. CONCLUSION

This paper uses a fast CU partitioning decision algorithm based on texture complexity and convolutional neural networks (CNN). It is well known that VVC uses QTMT segmentation, which improves the coding performance but also significantly increases the computational complexity. First, we perform statistics and analysis of CU segmentation patterns on the standard training set of videos, then process the size 64 × 64 CUs based on texture complexity, and then design a CNN model for CUs with multiple segmentation patterns in VVC to achieve fast segmentation. In the design of CNN model, we use symmetric convolution kernel and asymmetric convolution kernel in the convolution layer for better feature extraction. On the loss function, we use a cross-entropy function to train our model to improve the accuracy of the predictive partitioning model. Finally, by setting an appropriate threshold in the candidate list, the time spent in the encoding process can be reduced and the efficiency of the encoding process can be improved while maintaining high quality video. The experimental results show that, compared with the VTM10.0 anchoring algorithm, our fast scheme, in terms of coding time, decreases by 55.93% and BDBR increases by 1.81%; our moderate scheme, in terms of coding time, decreases by 47.9%. BDBR increases by only 1.29%.

REFERENCES

[1] T. Zhao, Y. Huang, W. Feng, Y. Xu, and S. Kwong, "Efficient VVC intra prediction based on deep feature fusion and probability estimation," *IEEE Trans. Multimedia*, early access, Sep. 21, 2022, doi: 10.1109/TMM.2022.3208516.

[2] W. Hamidouche, P. Philippe, S. A. Fezza, M. Haddou, F. Pescador, and D. Menard, "Hardware-friendly multiple transform selection module for the VVC standard," *IEEE Trans. Consum. Electron.*, vol. 68, no. 2, pp. 96–106, May 2022.

[3] M. Viitanen, J. Sainio, A. Mercat, A. Lemmetti, and J. Vanne, "From HEVC to VVC: The first development steps of a practical intra video encoder," *IEEE Trans. Consum. Electron.*, vol. 68, no. 2, pp. 139–148, May 2022.

[4] X. Wang and Y. Xue, "Fast HEVC intra coding algorithm based on Otsu's method and gradient," in *Proc. IEEE Int. Symp. Broadband Multimedia Syst. Broadcast. (BMSB)*, Jun. 2016, pp. 1–5.

[5] Q. Zhang, Y. Zhao, B. Jiang, and Q. Wu, "Fast CU partition decision method based on Bayes and improved de-blocking filter for H.266/VVC," *IEEE Access*, vol. 9, pp. 70382–70391, 2021.

[6] Y. Fan, H. Sun, J. Katto, and J. Ming, "A fast QTMT partition decision strategy for VVC intra prediction," *IEEE Access*, vol. 8, pp. 107900–107911, 2020.

[7] Q. Zhang, Y. Zhao, B. Jiang, L. Huang, and T. Wei, "Fast CU partition decision method based on texture characteristics for H.266/VVC," *IEEE Access*, vol. 8, pp. 203516–203524, 2020.

[8] 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 Trans. Circuits Syst. Video Technol.*, vol. 30, no. 6, pp. 1668–1682, Jun. 2020.

[9] W. Kuang, Y.-L. Chan, S.-H. Tsang, and W.-C. Siu, "Machine learning-based fast intra mode decision for HEVC screen content coding via decision trees," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 5, pp. 1481–1496, May 2020.

[10] J. Cui, T. Zhang, C. Gu, X. Zhang, and S. Ma, "Gradient-based early termination of CU partition in VVC intra coding," in *Proc. Data Compress. Conf. (DCC)*, Snowbird, UT, USA, Mar. 2020, pp. 103–112.

[11] M. Zhou, X. Wei, W. Jia, and S. Kwong, "Joint decision tree and visual feature rate control optimization for VVC UHD coding," *IEEE Trans. Image Process.*, vol. 32, pp. 219–234, 2023.

[12] X. Guan and X. Sun, "VVC fast ME algorithm based on spatial texture features and time correlation," in *Proc. Int. Conf. Digit. Soc. Intell. Syst. (DSInS)*, Chengdu, China, Dec. 2021, pp. 371–377.

[13] T. Li, M. Xu, R. Tang, Y. Chen, and Q. Xing, "DeepQTMT: A deep learning approach for fast QTMT-based CU partition of intra-mode VVC," *IEEE Trans. Image Process.*, vol. 30, pp. 5377–5390, 2021.

[14] Q. Zhang, Y. Wang, L. Huang, and B. Jiang, "Fast CU partition and intra mode decision method for H.266/VVC," *IEEE Access*, vol. 8, pp. 117539–117550, 2020.

[15] A. Tissier, W. Hamidouche, J. Vanne, F. Galpin, and D. Menard, "CNN oriented complexity reduction of VVC intra encoder," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Abu Dhabi, United Arab Emirates, Oct. 2020, pp. 3139–3143.

[16] M. Xu, T. Li, Z. Wang, X. Deng, R. Yang, and Z. Guan, "Reducing complexity of HEVC: A deep learning approach," *IEEE Trans. Image Process.*, vol. 27, no. 10, pp. 5044–5059, Oct. 2018.

[17] G. Tang, M. Jing, X. Zeng, and Y. Fan, "Adaptive CU split decision with pooling-variable CNN for VVC intra encoding," in *Proc. IEEE Vis. Commun. Image Process. (VCIP)*, Sydney, NSW, Australia, Dec. 2019, pp. 1–4.

[18] J. Zhao, A. Wu, B. Jiang, and Q. Zhang, "ResNet-based fast CU partition decision algorithm for VVC," *IEEE Access*, vol. 10, pp. 100337–100347, 2022.

[19] N. Tang, J. Cao, F. Liang, J. Wang, H. Liu, X. Wang, and X. Du, "Fast CTU partition decision algorithm for VVC intra and inter coding," in *Proc. IEEE Asia Pacific Conf. Circuits Syst. (APCCAS)*, Bangkok, Thailand, Nov. 2019, pp. 361–364.

[20] 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 *Proc. IEEE Vis. Commun. Image Process. (VCIP)*, Sydney, NSW, Australia, Dec. 2019, pp. 1–4.
[21] X. HoangVan, S. NguyenQuang, M. DinhBao, M. DoNgoc, and D. T. Duong, "Fast QTMT for H.266/VVC intra prediction using early-terminated hierarchical CNN model," in *Proc. Int. Conf. Adv. Technol. Commun. (ATC)*, Ho Chi Minh City, Vietnam, Oct. 2021, pp. 195–200.
[22] F. Galpin, F. Racape, S. Jaiswal, P. Bordes, F. Le Leannec, and E. Francois, "CNN-based driving of block partitioning for intra slices encoding," in *Proc. Data Compress. Conf. (DCC)*, Snowbird, UT, USA, Mar. 2019, pp. 162–171.
[23] Y. Li, Z. Liu, X. Ji, and D. Wang, "CNN based CU partition mode decision algorithm for HEVC inter coding," in *Proc. 25th IEEE Int. Conf. Image Process. (ICIP)*, Athens, Greece, Oct. 2018, pp. 993–997.
[24] X. Qian, Y. Zeng, W. Wang, and Q. Zhang, "Co-saliency detection guided by group weakly supervised learning," *IEEE Trans. Multimedia*, early access, Apr. 19, 2022, doi: 10.1109/TMM.2022.3167805.

HONGCHAN LI received the M.S. degree from the Sichuan University of Science and Engineering, Zigong, Sichuan, China, in 2010. Since 2010, she has been a Faculty Member with the School of Computer and Communication Engineering, Zhengzhou University of Light Industry, Zhengzhou, Henan, where she is currently an Associate Professor and a Master's Tutor. Her major research interests include cloud computation, intelligence information processing, computing intelligence, and data mining.

PENG ZHANG received the B.S. degree in electronic information engineering from the Henan Institute of Technology, Xinxiang, China. He is currently pursuing the master's degree in signal and information processing with the School of Computer and Communication Engineering, Zhengzhou University of Light Industry. His current research interests include image processing and multifunctional video coding.

BAOHUA JIN received the M.S. degree in computer science and technology from the Huazhong University of Science and Technology, Wuhan, China, in 2003. He is currently the Dean of the School of Computer and Communication Engineering, Zhengzhou University of Light Industry, China. His current research interests include big data and artificial intelligence.

QIUWEN ZHANG received the Ph.D. degree in communication and information systems from Shanghai University, Shanghai, China, in 2012. Since 2012, he has been a Faculty Member with the College of Computer and Communication Engineering, Zhengzhou University of Light Industry, where he is currently a Professor. He has published over 30 technical papers in the field of pattern recognition and image processing. His major research interests include 3D signal processing, machine learning, pattern recognition, video codec optimization, and multimedia communication.