information

Article

Extreme Learning Machine-Enabled Coding Unit Partitioning Algorithm for Versatile Video Coding

Xiantao Jiang [1,*], Mo Xiang [1], Jiayuan Jin [1] and Tian Song [2]

[1] Department of Information Engineering, Shanghai Maritime University, No. 1550, Haigang Ave., Shanghai 201306, China; 202130310123@stu.shmtu.edu.cn (M.X.); 202130310080@stu.shmtu.edu.cn (J.J.)
[2] Department of Electrical and Electronics Engineering, Tokushima University, 2-24, Shinkura-cho, Tokushima 770-8501, Japan; tiansong@ee.tokushima-u.ac.jp
* Correspondence: xtjiang@shmtu.edu.cn

Abstract: The versatile video coding (VVC) standard offers improved coding efficiency compared to the high efficiency video coding (HEVC) standard in multimedia signal coding. However, this increased efficiency comes at the cost of increased coding complexity. This work proposes an efficient coding unit partitioning algorithm based on an extreme learning machine (ELM), which can reduce the coding complexity while ensuring coding efficiency. Firstly, the coding unit size decision is modeled as a classification problem. Secondly, an ELM classifier is trained to predict the coding unit size. In the experiment, the proposed approach is verified based on the VVC reference model. The results show that the proposed method can reduce coding complexity significantly, and good image quality can be obtained.

Keywords: versatile video coding; coding unit; extreme learning machine; computation complexity

check for updates

Citation: Jiang, X.; Xiang, M.; Jin, J.; Song, T. Extreme Learning Machine-Enabled Coding Unit Partitioning Algorithm for Versatile Video Coding. *Information* **2023**, *14*, 494. https://doi.org/10.3390/info14090494

Academic Editor: Zoran H. Perić

Received: 7 July 2023
Revised: 31 August 2023
Accepted: 1 September 2023
Published: 7 September 2023

1. Introduction

With the rapid development of 5G, cloud computing, virtual reality, and other technologies, short video, live video, 4 K/8 K ultra-high-definition TV (UHDTV), virtual reality video, and other emerging video forms emerge in an endless stream, and gradually become people's production and life. An indispensable part of the video data traffic on the internet shows an explosive growth trend. At present, advanced video coding (AVC)/H.264 and high efficiency video coding (HEVC)/H.265 are used to encode videos on the internet. There is a large gap between coding performance and practical requirements. The new generation video coding standard VVC (versatile video coding)/H.266 was formally established in July 2020. This standard introduces a range of novel coding tools aimed at enhancing coding performance while maintaining consistent reconstruction quality. The outcome of these improvements has been a nearly 40% increase in coding efficiency [1]. It is important to note, however, that these enhancements in VVC have come at the cost of increased encoding complexity. The rapid surge in computational complexity has led to amplified hardware costs for VVC encoders and posed challenges for real-time video encoding applications. Current advancements in multimedia technologies and the evolution of virtual reality (VR) have led to a notable surge in the demand for images and videos featuring a 180/360-degree field of view (FoV) across diverse application domains [1]. A performance study of software and hardware encoding of omnidirectional 8 K video has been presented in [2]. Consequently, a significant research focus in the realm of video coding is dedicated to substantially reducing the computational complexity of VVC encoders, all the while minimizing potential compromises to video coding performance. In previous work [3], the saliency-based CU (coding unit) partitioning method was proposed to reduce intra-encoding complexity. However, in this work, the CU partitioning problem is modeled as a multi-classification problem, and a machine learning classifier is adopted to decide on the CU size.

The optimization of CU fast partitioning has remained a prominent focus within the video coding domain. Approaches addressing this challenge can be broadly categorized into traditional techniques and machine-learning-based methods. (1) Traditional methods: In the realm of knowledge-based methods, CU characteristics such as texture information and depth are key determinants of CU division. Biao et al. proposed an intra-CU partitioning algorithm employing global and local edge complexity, effectively minimizing encoding time [4]. A fast CU depth decision algorithm introduced by Min et al. utilizes hypothesis testing to statistically analyze CU depth and decide on optimal partitioning [5]. Sun et al. developed an efficient technique that leverages CU texture complexity to guide the division process, considering whether a CU should be further divided into sub-CUs [6]. An innovative approach by Fan et al. harnesses variance and gradient to expedite quadtree and multi-type tree (QTMT)-based partitioning, thus reducing computational load [7]. (2) Machine-learning-based methods: The second category embraces machine learning methods, encompassing decision trees, support vector machines (SVM), neural networks, and more, to expedite CU division. Among these, SVM has emerged as a widely adopted technique. Zhang et al. designed a rapid CU partitioning strategy leveraging an enhanced directed acyclic graph–support vector machine (DAG-SVM) classification model, framing the CU partitioning problem in H.266/VVC as a multi-class classification issue [8]. Cheng et al. introduced a decision-making algorithm based on SVM for CU division within VVC frames, utilizing entropy and texture contrast as indicators for division direction prediction [9]. Expanding beyond SVM, Tang et al. constructed a convolutional neural network (CNN) model integrating a variable pooling layer, which adapts fluidly to various CU shapes and predicts the necessity of CU division [10]. Fu et al. devised a VVC intra-coding algorithm that defers binary and ternary tree partitioning through Bayesian and rate-distortion (RD) cost methodologies [11]. Yang et al. proposed a low-complexity CTU partitioning approach, utilizing a decision tree to predict partitioning outcomes [12].

Recognizing the fast-paced nature of video coding research, we acknowledge the importance of incorporating recent works that have contributed to the evolving landscape of CU fast partitioning. Shang et al. use coding information to accelerate the coding process [13]. This method predicts the coding area of the current coding unit by analyzing neighboring CUs, reducing unnecessary splitting modes. Li et al. introduce a swift CU partitioning decision method that combines texture complexity and CNNs [14], and this method begins by analyzing CU segmentation patterns from the training set, processing large CU blocks based on texture complexity. Wang et al. develop a multi-stage network framework for addressing the given issue, segmenting CUs into stages based on block size and adaptively extracting relevant features [15]. Tissier et al. propose a decision tree (DT) model to predict probable splits for each block [16]. Based on this prediction, the encoder’s rate-distortion process focuses on the N most likely splits. Shang et al. propose a rapid CU size decision algorithm to enhance intra-coding efficiency [17], incorporating coding and texture data. The method includes a swift quadtree decision approach (FQD) and a prompt multi-type tree decision method (FMD). Zhang et al. propose an efficient partition algorithm by introducing the empirical variogram along with Mahalanobis distance to quantify the disparity between the horizontal and vertical directions of the coding unit [18].

There are some problems existing in the past methods. (1) The traditional statistical method has limited ability to reduce the coding complexity, and it does not strike a good balance between coding efficiency and coding computational complexity [19]. (2) Machine learning methods need to extract image features [20], which brings additional overhead. In this work, an efficient extreme learning machine-based CU size decision method is proposed to reduce the coding complexity for VVC inter-prediction. The main contributions of this paper are:

(1) In this work, we propose a distinctive approach by modeling the CU size decision as a classification problem and employing the extreme learning machine model for predicting the CU partitioning mode. The uniqueness lies in the utilization of the ELM model, which offers the advantage of predicting the CU partitioning mode without

necessitating image feature extraction. This stands in contrast to traditional machine learning algorithms and enhances efficiency.

(2) Additionally, to further elevate the predictive accuracy, we incorporate an online learning method. By continuously adapting to the evolving dataset, this technique improves the prediction accuracy of our proposed approach.

The remainder of this paper is organized as follows. The background details are introduced in Section 2. The proposed algorithm is presented in Section 3. Experimental results are shown in Section 4. Concluding remarks are given in Section 5.

2. Technical Background

2.1. VVC/H.266

VVC, also known as H.266, is a new video codec standard that was released by the Joint Video Experts Team (JVET), which is designed to improve upon the previous H.265/HEVC standard by offering even better compression and reducing the amount of data required to store high-quality video files. In terms of encoding computational complexity, VVC is more complex than its predecessor H.265. This enhances its coding efficiency but at the cost of computational demand, and thus, it may require specialized hardware to meet efficient encoding/decoding requirements. Some of the key technologies and tools in VVC include: (1) Enhanced coding tools: VVC introduces various enhanced coding tools, such as larger block sizes for inter-prediction (up to 128 $\times$ 128), variable block sizes for intra-prediction (down to 4 $\times$ 4), and support for multiple reference pictures. (2) Prediction tools: VVC improves inter-prediction with advanced tools such as bi-directional prediction, refined motion vector prediction, and fractional-pel motion compensation. These tools enhance the accuracy of motion compensation and prediction. (3) Transform tools: VVC introduces new transform tools, such as the directional discrete cosine transform (DCT), which adapts to the directional characteristics of image content, improving compression efficiency. (4) Entropy coding: context-adaptive binary arithmetic coding (CABAC), a sophisticated entropy coding technique, is further enhanced in VVC, allowing for more efficient representations of syntax elements.

The introduction of binary trees and ternary trees greatly increased the complexity of partitioning coding blocks [21]. The partition method of nesting multi-type trees in the quadtree of VVC significantly improves the coding performance, making the partition more flexible and no longer limited to squares. It also adds rectangular blocks. VVC still uses the coding tree unit (CTU) structure in HEVC, and the maximum size of luma blocks is expanded to 128 $\times$ 128 with no-splitting (NS). CTU is first partitioned by quadtree (QT), and then can be further partitioned using multi-type tree (MTT). The multi-type tree structure includes four types of partitioning methods, as shown in Figure 1, which are vertical binary tree partitioning (BTV), horizontal binary tree partitioning (BTH), vertical ternary partitioning (TTV), and horizontal ternary tree partitioning (TTH). In VVC, the maximum size allowed for the root node of the quadtree is 128 $\times$ 128, and the minimum size allowed for the leaf nodes of the quadtree is 16 $\times$ 16.

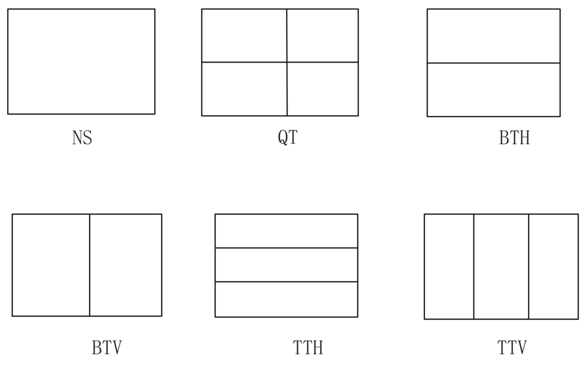

Figure 1. CU partitioning modes.

2.2. Extreme Learning Machine

Extreme learning machine (ELM) is the recent machine learning algorithm that has gained popularity in the industry and academia. It is a fast and efficient algorithm used for regression, classification, and clustering problems. ELM is a single-hidden-layer feedforward neural network in which the input weights and biases are randomly initialized, and only the output weights are learned during the training process. The motivation behind the development of the ELM algorithm was to overcome the limitations of traditional neural networks (NNs). Traditional NNs require a considerable amount of time and effort to obtain optimized weights, and the training process is computationally expensive. On the other hand, ELM is a fast-learning algorithm that uses random weights, which makes it less prone to over-fitting and easier to train. The ELM architecture consists of an input layer, a hidden layer, and an output layer. The input layer receives a set of features from the data, and the hidden layer transforms these data into a higher-dimensional space. The output layer generates the predicted output value based on the weights calculated during the training process.

Extreme learning machines are a type of machine learning algorithm that has been used for various multimedia applications, including signal coding. In signal coding, ELMs can be trained to learn compressed representations of multimedia signals such as images or video [22,23]. In this work, the ELM-enabled coding unit partitioning algorithm is proposed to reduce the computational complexity of VVC inter-prediction.

3. The Proposed Extreme Learning Machine-Enabled Coding Unit Partitioning Algorithm

3.1. Algorithm Model

In this work, the CU partitioning is modeled as a multi-class problem, and there are six partitioning modes: no-splitting (NS), quadtree (QT), vertical binary tree partitioning (BTV), horizontal binary tree partitioning (BTH), vertical ternary partitioning (TTV), and horizontal ternary tree partitioning (TTH). The feature of CU mode partitioning is $x_i = \{x_{i1}, x_{i2}, x_{i3}\}$, which is composed of the distortion cost of CU rate, CU depth, and CU prediction residual.

The ELM algorithm is an improved algorithm based on single-hidden-layer feedforward neural network (SLFN). The biggest advantage of the ELM algorithm over the traditional SLFN algorithm is that it does not require updating the parameters in training, such as the weight between the input layer and the hidden layer and the threshold of the hidden layer neurons. Once the number of hidden layer nodes is determined, the ELM algorithm can directly generate the weights and thresholds between the nodes randomly, thus greatly simplifying the process and time of network training. Figure 2 shows the structure of the ELM, with the ELM neuron settings as follows: input layer: n nodes, hidden layer neurons: L nodes, output layer neurons: m nodes. In this work, CU partitioning is modeled as a multi-classification problem, and the CU modes are divided into six categories. Therefore, the value of the parameter m is set to 6. For each sample, the input is $x = [x_1, x_2, \ldots, x_n]^T$, b is the bias of the hidden layer nodes, b_o is the bias of the output layer nodes, $g(.)$ is the activation function of the hidden layer, and the output function of the output layer is a linear function. The output of the output layer is $y = [y_1, y_2, \ldots, y_m]^T$, and the expected output is $t = [t_1, t_2, \ldots, t_m]^T$. Let $w = w_{ij}$ denote the connection weights between input layer neurons and hidden layer neurons, and let $\beta = \{\beta_{ij}\}$ denote the connection weights between hidden layer neurons and output layer neurons, let $h(x)$ denote the output of all neurons in the hidden layer for input x, and the expression for $h(x)$ is:

$$h(x) = [g(w_1 x + b_1) g(w_2 x + b_2) \ldots g(w_l x + b_l)] \ . \tag{1}$$

Then, the output y is computed as

$$y = h(x)\beta_i + b_o \ . \tag{2}$$

The error ξ_i between the actual output and the expected output of the network can be minimized as much as possible to satisfy the equation:

$$\lim_{L\to\infty} || \sum_{i=1}^{L} h(x_i)\beta - t_i|| = 0\ , \tag{3}$$

where $||.||$ denotes the norm, that is, there exist appropriate β_i, w_i, and b_i that satisfy the following equation:

$$H\beta = T\ , \tag{4}$$

where $H = [h(x_1), h(x_2), \ldots, h(x_N)]^T$ is the output matrix of the hidden layer, and $T = [t_1, t_2, \ldots, t_N]^T$ is the expected output.

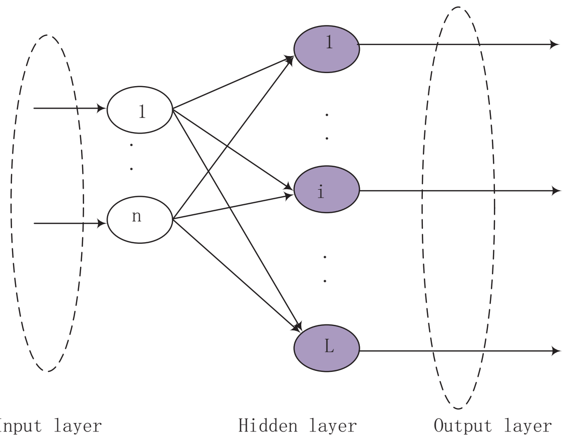

Figure 2. ELM structure.

According to statistical learning theory, empirical risk and structural risk constitute the actual risk. Therefore, the weight and actual error in the output are minimized, i.e., the minimization of $||H\beta - T||$ and $||\beta||$, respectively:

$$\min: L_P = \frac{1}{2}||\beta|| + \frac{1}{2}C||\xi_i||\ , \tag{5}$$

$$s.t.: h(x_i)\beta = t_i^T - \xi_i^T, i = 1, 2, \ldots, N\ , \tag{6}$$

where C is a constant, and ξ_i represents the error between the actual value and the expected value of the input set x_i. According to KKT theory, training ELM is equivalent to solving the following dual optimization problem:

$$L_D = \frac{1}{2}||\beta|| + \frac{1}{2}C||\xi_i|| - \sum_{i=1}^{N}\sum_{j=1}^{m} \alpha_{ij}(h(x_i)\beta_j - t_{ij} + \xi_{ij})\ , \tag{7}$$

where α_{ij} is a positive Lagrange multiplier, $\alpha_i = [\alpha_{i1}, \alpha_{i2}, \ldots, \alpha_{im}]^T$, and $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_N]^T$. If the output weights are minimized and the error is minimized, then:

$$\frac{\partial L_D}{\partial \beta_j} = 0 \to \beta = H^T\alpha\ , \tag{8}$$

$$\frac{\partial L_D}{\partial \xi_i} = 0 \to \alpha_i = C\xi_i\ , \tag{9}$$

$$\frac{\partial L_D}{\partial \alpha_i} = 0 \to h(x_i)\beta - t_i^T + \xi_i^T\ . \tag{10}$$

Formula (10) can be represented using matrix methods as

$$(\frac{1}{C} + HH^T)\alpha = T\ . \tag{11}$$

According to Equations (8) and (11), we can obtain:

$$\beta = H^T(\frac{1}{C} + HH^T)^{-1}T\ . \tag{12}$$

The output of the ELM classifier can be represented as follows:

$$f(x) = h(x_i)\beta = h(x_i)H^T(\frac{1}{C} + HH^T)^{-1}T\ . \tag{13}$$

3.2. The Overall Framework

This work proposes the constructed extreme learning machine (ELM) multi-classifier to divide CU modes. First, the classifier is trained on the first frame of each group of pictures (GOPs), and then the CU mode is determined in the remaining frames of the GOPs. The GOPs size used in our experiments was 32. Moreover, little time is required for the training. The reasons for this are that (1) ELM is a single-layer feedforward neural network that trains rapidly. Compared to traditional iterative training algorithms, ELM requires a single weight initialization and output layer weight computation, resulting in faster training times. (2) ELM has low memory requirements, as it only needs to store weights and biases between the input and hidden layers. This makes ELM suitable for large-scale datasets and resource-constrained applications. Therefore, the impact of the time required for the training on experimental results is small.

In the process of CU mode partition, first of all, the cost function of the CU rate distortion is calculated, and the corresponding texture complexity of the CU and the prediction residual feature of the CU are then used to partition the modes using the pre-trained ELM classifier. Let $f_j(x)$ denote the output function of the jth output node, $f(x) = [f_1(x), \ldots, f_m(x)]^T$, then the CU mode labeling can be expressed as follows:

$$label(x) = \arg\max_m f_i(x)\ . \tag{14}$$

The overall flowchart of the ELM-enabled CU partitioning algorithm is shown in Figure 3. The proposed method can be divided into three steps:

(1) Feature extraction and parameter learning: As shown in Figure 4, in the CU partitioning process, the first frame of each group of pictures (GOPs) is designated as the parameter learning frame. For each coding unit (CU), an effective feature vector is extracted, which forms a training set. During this step, the number of hidden layer nodes and the activation function are configured.

(2) ELM network initialization and training: The ELM network is initialized by generating random weights (w) and biases (b). The hidden layer output matrix (H) is computed, and using this matrix, the weight matrix of the output layer is calculated. This leads to the establishment of the output function of the ELM network.

(3) CU mode partitioning using ELM classifier: Building upon the ELM classifier's output function, CU mode partitioning is conducted on the remaining frames of the GOPs. This process results in the determination of the final CU partitioning mode, thereby enabling an efficient encoding strategy for subsequent frames.

Through these three interconnected steps, the ELM-enabled CU partitioning algorithm effectively leverages the learned features and the classification capability of the ELM network to optimize the CU partitioning decisions for enhanced video coding performance.

The proposed method can accelerate the CU size decision, significantly. Moreover, the ELM-based partitioning approach is implemented across all levels of CTU partitioning. This method optimizes partition size determination within the coding tree, ensuring consistent application throughout various partitioning levels.

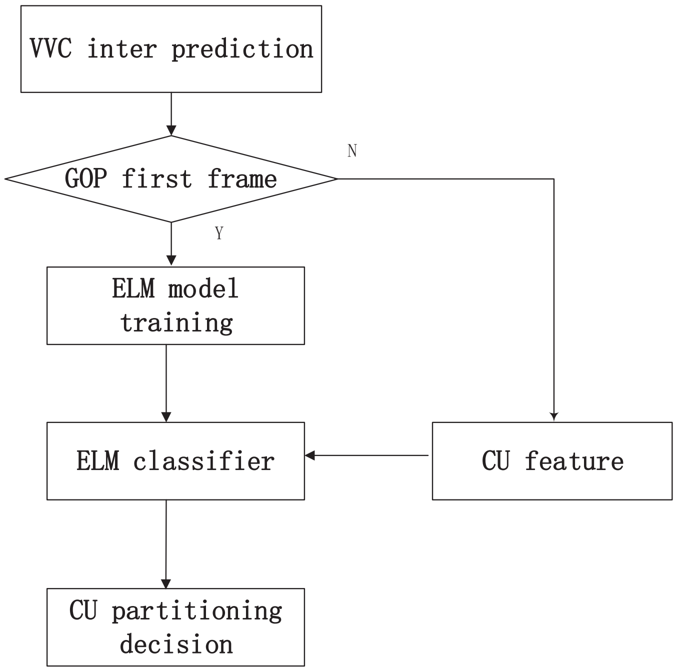

Figure 3. ELM-enabled CU partitioning algorithm.

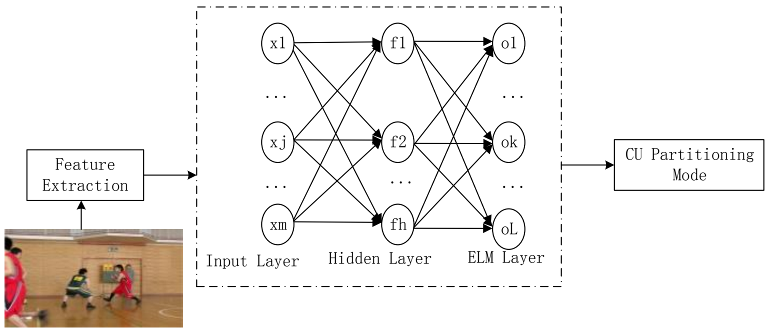

Figure 4. The ELM-based machine learning structure.

4. Experimental Results

To evaluate the performance of the proposed ELM-based VVC encoder, this section shows the experimental results obtained when implementing the proposed algorithm with the H.266/VVC reference software (https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM accessed on 1 August 2023). The random-access and low-delay configurations are used as the configuration files, with a QP of {22, 27, 32, 37}. The maximum CTU size is 128 × 128. The simulation environments are shown in Table 1. Testing on a large scale requires resources and time. Given the high resolution and complexity of Class A (4096 × 2160) sequences, testing them may require more computational resources and time than our research plan. Therefore, with limited resources and time, the class A sequences were not tested in this work. The Bjontegaard delta bit rate (BDBR) is used to assess the coding efficiency. We adopt PSNR and MS-SSIM [24] for evaluating the video quality, which are commonly used metrics in video compression. Moreover, the time saving (TS) is used to measure the reduction in computational complexity of the VVC encoder, which is calculated as

$$TS = \frac{1}{4} \times \sum_{i=1}^{4} \frac{\text{Time}_{VTM12.0}(QP_i) - \text{Time}_{\text{proposed}}(QP_i)}{\text{Time}_{VTM12.0}(QP_i)} \times 100\%\,, \tag{15}$$

where $Time_{VTM12.0}(QP_i)$ and $Time_{proposed}(QP_i)$ denote the encoding time of using VTM12.0 and the proposed algorithm with different QP. Furthermore, in the VTM12.0, the two configuration profiles, including "encoder_randomaccess_vtm.cfg" and "encoder_lowdelay_vtm.cfg", can be chosen to verify the algorithm's performance. During the experiment, we selected different video sequences to test the encoding performance with different QP values. The operating system is Windows 7, and the processor is Inter(R) Core i3-2310M with 8 GB memory. The language used is the Python3 version for learning. The C++ version of Pytorch is used to load the model.

Table 1. The system parameters configurations.

Item	Description
Software	VTM12.0
Video size	1920 × 1080, 1280 × 720, 832 × 480, 416 × 240
Configurations	Random access (RA), low delay (LD)
QP	22, 27, 32, 37
Maximum CTU size	128 × 128

According to the algorithmic performance under random-access (RA) mode in Table 2, the average BDBR has increased by 0.76% and the computation time has been reduced by 50.04%. In the low-delay (LD) mode shown in Table 2, the average BDBR has increased by 0.59% and the time has been reduced by 50.19%. The proposed method can reduce the encoding complexity with less than 1% encoding efficiency loss for the low-delay (LD) profile and random-access (RA) profile, which can achieve a significant reduction in encoding complexity and maintain high encoding efficiency. Furthermore, it can strike a balance between encoding complexity and encoding efficiency. For the BasketballDrive sequence, the encoding efficiency loss is 2.17% for the RA profile. There are two reasons for this result: (1) Resolution is one of the important factors that affect the encoding effects. Higher-resolution video sequences typically have more detail and information, which can lead to larger data volumes and higher bit rates during the encoding process. (2) The degree of image motion in the video also has a significant impact on the encoding effect. Highly moving image sequences may cause more displacement and motion estimation errors, resulting in a reduced coding effect. In contrast, the corresponding video sequence BasketballDrill has a lower resolution and flat image motion, so the encoding efficiency loss in the BasketballDrill sequence is only 0.22% for RA profile.

Table 2. Experimental results of ELM-enabled CU partitioning algorithm.

		RA		LD	
Size	**Sequence**	**BDBR (%)**	**TS (%)**	**BDBR (%)**	**TS (%)**
1920 × 1080	BasketballDrive	2.17	46.29	1.59	41.69
	BQTerrace	0.33	52.03	0.22	50.01
	Cactus	0.69	51.66	0.64	57.64
	Kimono	1.51	50.14	1.03	53.81
	ParkScene	0.62	55.47	0.46	52.56
832 × 480	BasketballDrill	0.22	52.06	0.20	54.79
	BQMall	0.40	54.25	0.49	54.20
	PartyScene	0.27	46.86	0.16	45.75
	RaceHorsesC	1.51	41.88	0.86	43.97
416 × 240	BlowingBubbles	0.66	41.49	0.72	41.22
	BQSquare	0.35	43.19	0.10	46.82
	RaceHorses	1.71	40.96	1.04	44.24
1280 × 720	FourPeople	0.27	61.57	0.32	58.59
	KristenAndSara	0.65	63.27	0.70	57.35
Average		0.76	50.04	0.59	50.19

To obtain more direct feedback on the quality of the proposed algorithm, we compare the rate–distortion curves of the algorithm presented with those of the original testing framework, and the results are shown in Figure 5. The rate–distortion curves of the proposed algorithm are compared under the best-case and worst-case scenarios of the test sequences in the RA profile. Therefore, this comparison enables a comprehensive evaluation of the algorithm's performance across different scenarios. It can be observed that even under these extreme conditions, the rate–distortion curves of the proposed algorithm are almost identical to each other. We can see that the proposed method can significantly reduce the coding time, while the RD curve is almost the same as the VVC reference model. This indicates the proposed method can reduce the computational complexity with negligible RD performance loss. For the R-D curves regarding MS-SSIM in Figure 6, we fine-tune our networks with the MS-SSIM loss. The MS-SSIM results show a similar trend to that of PSNR. Moreover, the visual comparison for the BasketballDrive sequence is shown in Figure 5. It can be seen that, compared with the original image, the quality of the reconstructed image does not decrease significantly (see in Figure 7).

Furthermore, the performance of the ELM-enabled CU partitioning algorithm is compared to previous works [13,14,25–29], and the results are shown in Table 3. Shang et al.'s [13] work is based on coding information. Tissier et al.'s [28] work uses a machine learning classifier. Yeo et al.'s work [25] is based on a CNN classifier. Amestoy et al.'s method [26] is based on a lightweight machine learning approach. Pan et al.'s [27] method is based on multi-domain information. Li et al.'s [14] work is based on a CNN classifier. Tang et al.'s [29] work is based on temporal correlation information.

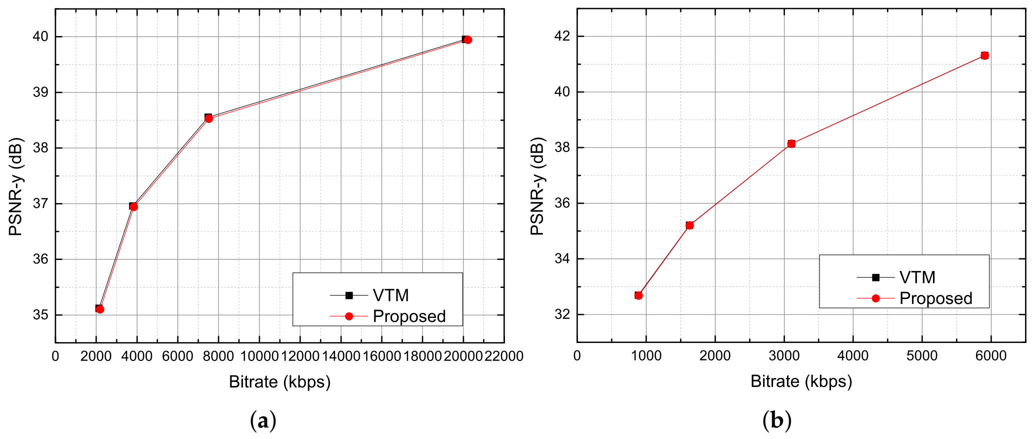

Figure 5. R-D curve. (**a**) BasketballDrive (RA); (**b**) BasketballDrill (RA).

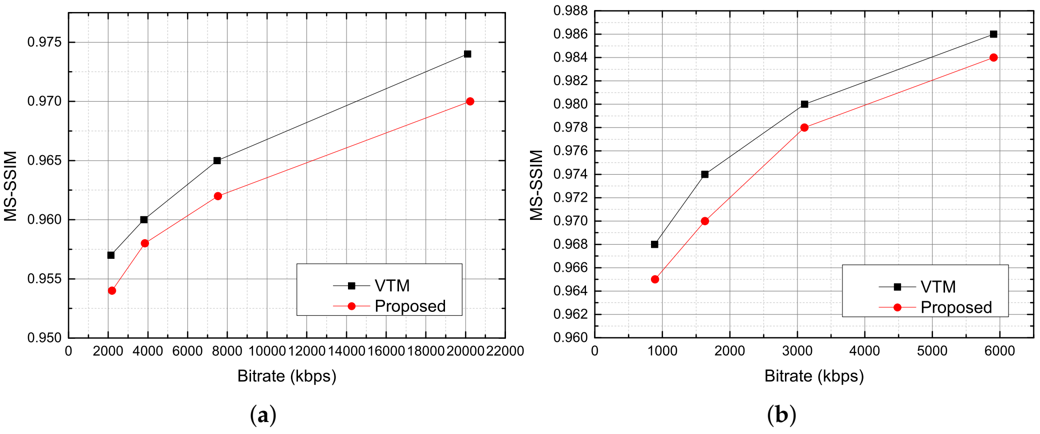

Figure 6. MS-SSIM curve. (**a**) BasketballDrive (RA); (**b**) BasketballDrill (RA).

Figure 7. Visual comparison for BasketballDrive sequence. (**a**) Original image; (**b**) reconstructed image.

It can be seen from the comparison of the experimental results that the computation complexity of the proposed method can be reduced significantly; meanwhile, the encoding efficiency of this method is better than previous work.

Table 3. The performance of the proposed method compared with previous work.

	Method	**(BDBR, TS)**
RA	Proposed	(0.76, 50.04)
	Yeo's [25]	(1.10, 13.10)
	Amestoy's [26]	(0.61, 30.10)
	Shang's [13]	(1.56, 40.08)
	Tissier's [28]	(1.65, 31.3)
LD	Proposed	(0.59, 50.19)
	Pan's [27]	(2.52, 24.83)
	Li's [14]	(1.29, 47.90)
	Tang's [29]	(1.34, 31.43)

5. Conclusions

In this work, the extreme learning machine-based coding unit partitioning algorithm of multimedia signal coding is proposed to reduce the computational complexity. An online learning method is adopted to train the ELM classifier model. The simulation results show that the ELM-enabled algorithm can significantly reduce the computational complexity of the VVC encoder. In the future, a parallel approach will be explored to accelerate the encoder.

Author Contributions: X.J. designed the algorithm, conducted all experiments, analyzed the results, and wrote the manuscript. M.X. wrote the manuscript. J.J. conceived the algorithm and wrote the manuscript. T.S. conducted the literature review and wrote the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This work is sponsored by Shanghai Pujiang Program (22PJD028).

Data Availability Statement: Data available on request due to restrictions, e.g., privacy or ethical.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Polak, L.; Kufa, J.; Kratochvil, T. On the Compression Performance of HEVC, VP9 and AV1 Encoders for Virtual Reality Videos. In Proceedings of the IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), Paris, France, 27–29 October 2020; pp. 1–5.
2. Kufa, J.; Polak, L.; Simka, M.; Stech, A. Software and Hardware Encoding of Omnidirectional 8 K Video: A Performance Study. In Proceedings of the 33rd International Conference Radioelektronika, Pardubice, Czech Republic, 19–20 April 2023; pp. 1–5.
3. Li, W.; Jiang, X.; Jin, J.; Song, T.; Yu, F.R. Saliency-Enabled Coding Unit Partitioning and Quantization Control for Versatile Video Coding. *Information* **2022**, *13*, 394. [CrossRef]
4. Min, B.; Ray, C.C. A fast CU size decision algorithm for the HEVC intra encoder. *IEEE Trans. Circuits Syst. Video Technol.* **2014**, *25*, 892–896.
5. Jimenez-Moreno, A.; Martínez-Enríquez, E.; Díaz-de-María, F. Bayesian adaptive algorithm for fast coding unit decision in the High Efficiency Video Coding (HEVC) standard. *Signal Process. Image Commun.* **2017**, *56*, 1–11. [CrossRef]
6. Sun, X.; Chen, X.; Xu, Y.; Xiao, Y.; Wang, Y.; Yu, D. Fast CU size and prediction mode decision algorithm for HEVC based on direction variance. *J. Real-Time Image Process.* **2019**, *16*, 1731–1744. [CrossRef]
7. Fan, Y.; Chen, J.; Sun, H.; Katto, J.; Jing, M. A fast QTMT partition decision strategy for VVC intra prediction. *IEEE Access* **2020**, *8*, 107900–107911. [CrossRef]
8. Zhang, Q.; Wang, Y.; Huang, L.; Jiang, B.; Wang, X. Fast CU partition decision for H.266/VVC based on the improved DAG-SVM classifier model. *Multimed. Syst.* **2021**, *27*, 1–14. [CrossRef]
9. Chen, F.; Ren, Y.; Peng, Z.; Jiang, G.; Cui, X. A fast CU size decision algorithm for VVC intra prediction based on support vector machine. *Multimed. Tools Appl.* **2020**, *79*, 27923–27939. [CrossRef]
10. Tang, G.; Jing, M.; Zeng, X.; Fan, Y. Adaptive CU split decision with pooling-variable CNN for VVC intra encoding. In Proceedings of the IEEE Visual Communications and Image Processing (VCIP), Sydney, NSW, Australia, 1–4 December 2019; pp. 1–4.
11. Fu, T.; Zhang, H.; Mu, F.; Chen, H. Fast CU partitioning algorithm for H.266/VVC intra-frame coding. In Proceedings of the IEEE International Conference on Multimedia and Expo (ICME), Shanghai, China, 8–12 July 2019; pp. 55–60.
12. Yang, H.; Shen, L.; Dong, X.; Ding, Q.; An, P.; Jiang, G. Low-complexity CTU partition structure decision and fast intra mode decision for versatile video coding. *IEEE Trans. Circuits Syst. Video Technol.* **2019**, *30*, 1668–1682. [CrossRef]
13. Shang, X.; Li, G.; Zhao, X.; Zuo, Y. Low complexity inter coding scheme for Versatile Video Coding (VVC). *J. Vis. Commun. Image Represent.* **2023**, *90*, 103683. [CrossRef]
14. Li, H.; Zhang, P.; Jin, B.; Zhang, Q. Fast CU Decision Algorithm Based on Texture Complexity and CNN for VVC. *IEEE Access* **2023**, *11*, 35808–35817. [CrossRef]
15. Wang, Y.; Liu, Y.; Zhao, J.; Zhang, Q. Fast CU Partitioning Algorithm for VVC Based on Multi-Stage Framework and Binary Subnets. *IEEE Access* **2023**, *11*, 56812–56821. [CrossRef]
16. Tissier, A.; Hamidouche, W.; Mdalsi, S.B.D.; Vanne, J.; Galpin, F.; Menard, D. Machine Learning Based Efficient QT-MTT Partitioning Scheme for VVC Intra Encoders. *IEEE Trans. Circuits Syst. Video Technol.* **2023**, *33*, 4279–4293. [CrossRef]
17. Shang, X.; Li, G.; Zhao, X.; Han, H.; Zuo, Y. Fast CU size decision algorithm for VVC intra coding. *Multimed. Tools Appl.* **2023**, *82*, 28301–28322. [CrossRef]
18. Zhang, M.; Hou, Y.; Liu, Z. An early CU partition mode decision algorithm in VVC based on variogram for virtual reality 360 degree videos. *EURASIP J. Image Video Process.* **2023**, *1*, 9. [CrossRef]
19. Saldanha, M.; Sanchez, G.; Marcon, C.; Agostini, L. Configurable Fast Block Partitioning for VVC Intra Coding Using Light Gradient Boosting Machine. *IEEE Trans. Circuits Syst. Video Technol.* **2022**, *32*, 3947–3960. [CrossRef]
20. Jin, D.; Lei, J.; Peng, B.; Li, W.; Ling, N.; Huang, Q. Deep Affine Motion Compensation Network for Inter Prediction in VVC. *IEEE Trans. Circuits Syst. Video Technol.* **2022**, *32*, 3923–3933. [CrossRef]
21. Huang, Y.W.; An, J.; Huang, H.; Li, X.; Hsiang, S.T.; Zhang, K.; Gao, H.; Ma, J.; Chubach, O. Block Partitioning Structure in the VVC Standard. *IEEE Trans. Circuits Syst. Video Technol.* **2021**, *31*, 3818–3833. [CrossRef]
22. Gaspar, A.; Oliva, D.; Hinojosa, S.; Aranguren, I.; Zaldivar, D. An optimized Kernel Extreme Learning Machine for the classification of the autism spectrum disorder by using gaze tracking images. *Appl. Soft Comput.* **2022**, *120*, 108654. [CrossRef]
23. Ganesan, A.; Santhanam, S.M. A novel feature descriptor based coral image classification using extreme learning machine with ameliorated chimp optimization algorithm. *Ecol. Inform.* **2022**, *68*, 101527. [CrossRef]
24. Wang, Z.; Simoncelli, E.P.; Bovik, A.C. Multiscale structural similarity for image quality assessment. In Proceedings of the Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, Pacific Grove, CA, USA, 9–12 November 2003; pp. 1398–1402.
25. Yeo, W.H.; Kim, B.G. CNN-based fast split mode decision algorithm for versatile video coding (VVC) inter prediction. *J. Multimed. Inf. Syst.* **2021**, *8*, 147–158. [CrossRef]
26. Amestoy, T.; Mercat, A.; Hamidouche, W.; Menard, D.; Bergeron, C. Tunable VVC frame partitioning based on lightweight machine learning. *IEEE Trans. Image Process.* **2019**, *29*, 1313–1328. [CrossRef] [PubMed]
27. Pan, Z.; Zhang, P.; Peng, B.; Ling, N.; Lei, J. A CNN-based fast inter coding method for VVC. *IEEE Signal Process. Lett.* **2021**, *28*, 1260–1264. [CrossRef]

28. Tissier, A.; Hamidouche, W.; Vanne, J.; Menard, D. Machine Learning Based Efficient Qt-Mtt Partitioning for VVC Inter Coding. In Proceedings of the IEEE International Conference on Image Processing (ICIP), Bordeaux, France, 16–19 October 2022; pp. 1401–1405.
29. Tang, N.; Cao, J.; Liang, F.; Wang, J.; Liu, H.; Wang, X.; Du, X. Fast CTU partition decision algorithm for VVC intra and inter coding. In Proceedings of the IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), Bangkok, Thailand, 11–14 November 2019; pp. 361–364.