Hindawi
Mathematical Problems in Engineering
Volume 2022, Article ID 1440323, 11 pages
https://doi.org/10.1155/2022/1440323

Research Article

Transformer-Based Data-Driven Video Coding Acceleration for Industrial Applications

Yixiao Li,[1,2] Lixiang Li,[1,2] Zirui Zhuang,[3] Yuan Fang,[1,2] Haipeng Peng,[1,2] and Nam Ling[4]

[1]*Information Security Center, State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China*
[2]*National Engineering Laboratory for Disaster Backup and Recovery, Beijing University of Posts and Telecommunications, Beijing 100876, China*
[3]*State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China*
[4]*Department of Computer Science and Engineering, Santa Clara University, 95053 Santa Clara, USA*

Correspondence should be addressed to Lixiang Li; lixiang@bupt.edu.cn

Received 14 March 2022; Revised 4 September 2022; Accepted 14 September 2022; Published 27 September 2022

Academic Editor: Ardashir Mohammadzadeh

With the exploding development of edge intelligence and smart industry, deep learning-based intelligent industrial solutions are promptly applied in the manufacturing process. Many intelligent industrial solutions such as automatic manufacturing inspection are computer vision based and require fast and efficient video encoding techniques so that video streams can be processed as quickly as possible either at the edge cluster or over the cloud. As one of the most popular video coding standards, the high efficiency video coding (HEVC) standard has been applied to various industrial scenes. However, HEVC brings not only a higher compression rate but also a significant increase in encoding complexity, which hinders its practical application in industrial scenarios. Fortunately, a large amount of video coding data makes it possible to accelerate the encoding process in the industry. To speed up the video coding process in some industrial scenes, this paper proposes a data-driven fast approach for coding tree unit (CTU) partitioning in HEVC intracoding. First, we propose a method to represent the partition result of a CTU as a column vector of length 21. Then, we employ lots of encoding data produced in normal industry scenes to train transformer models used to predict the partitioning vector of the CTU. Finally, the final partitioning structure of the CTU is generated from the partitioning vector after a postprocessing operation and used by an industrial encoder. Compared with the original HEVC encoder used by some industrial applications, experiment results show that our approach achieves 58.77% encoding time reduction with 3.9% bit rate loss, which indicates that our data-driven approach for video coding has great capacity working in industrial applications.

1. Introduction

With the development of the smart industry, image and video play a key role in many industrial scenarios [1–3]. As a result, video coding standards have been used more widely than ever [4]. Although advanced video coding (AVC) was introduced in 2003, it is still used by many applications today due to its fast coding speed [5]. However, with the increasing demand for high resolution and ultra-high resolution video and the improvement of hardware computing, HEVC is gradually replacing AVC because of its superior coding efficiency. Although high efficiency video coding (HEVC) was developed about one decade ago, the computational complexity is still not low at present due to the introduction of many advanced coding tools [6]. Some certain industrial scenarios have an urgent demand for real-time high definition display, and high encoding complexity hinds the applications of HEVC in these scenarios. Fortunately, it is becoming more mature for the use of big data and deep learning techniques in the industry [7–12]. Besides, mobile video applications have exploded in recent years and produced massive encoding data, which makes it possible for us

to employ big data and deep learning techniques to reduce the encoding time and speed up the encoding process in industrial applications [13].

The main computing burden of HEVC encoding comes from a complicated partition rule for intracoding called quadtree structure. To lower the computational complexity, many scholars have developed lots of fast algorithms for HEVC encoding by using various techniques, such as the learning-based technique and 3-type fuzzy logic system (FLS) [14]. Learning-based techniques are good at finding patterns in data. Dong et al. [15] proposed a learning-based fast algorithm for versatile video coding (VVC) from two aspects of mode selection and prediction terminating to reduce coding complexity, and 3-type fuzzy logic systems are good at solving equations or finding a mathematical model to represent the relationship between output and associated input variables [16]. With the advances in modeling problems, 3-type FLS is suitable for mode decision or rate control tasks in video coding and may bring potential improvement for encoding performance [14]. There are quite a number of algorithms focusing on fast coding unit (CU) partitioning. Furthermore, with the popularity of deep learning technology [17], researchers use neural networks (NNs) to boost CU partitioning and achieve satisfactory results [18]. Works using the deep learning method can be roughly classified into two main categories which are the multistage partition approach and the end-to-end structure decision approach.

In the category of multistage partition, approaches regard the structure determination of CTU partitioning as a combination of several binary classification problems. Xu et al. [19] proposed a three-level CTU splitting algorithm based on a convolution neural network (CNN). They trained three deep CNN models, of which each predicted the split flag for a CU in a certain depth. Kim and Ro [20] also proposed a CTU partition algorithm by using three CNN-based classifiers, they used different networks to predict the splitting decisions of CUs in different sizes. Shi et al. [21] proposed an AK-CNN for fast CTU partition prediction. Their AK-CNN classifiers are well-designed and can detect texture complexity of the CU quickly. Shen et al. [22] proposed early determination and a bypass strategy for CU size decisions by using the texture property of the current CU and coding information from neighboring CUs. Moreover, Shen et al. [23] proposed a fast intermode decision algorithm for HEVC by jointly using the interlevel correlation of quadtree structure and the spatiotemporal correlation. Based on visual perception and machine learning, Chen et al. [24] proposed a fast algorithm by using random forest models to quickly select the partition for VVC intracoding. However, methods in this category usually need as many as three NN models, which require much training work and cause implementation difficulties. Besides, the splitting of the CU is usually related to partition statuses of neighboring CUs, and methods in this category obviously ignore the splitting information from neighboring CUs by considering the splitting problem hierarchically.

In the category of end-to-end decision, one partition structure or several possible result candidates of the CTU can be generated through a single prediction. Liu et al. [18] proposed a VLSI friendly approach to partition a CTU by designing a shallow CNN. Feng et al. [25] proposed a fast block partitioning algorithm using CNN-based depth map prediction for HEVC intracoding. They used a depth map to represent the partition structure of the CTU so that quadtree structure can be predicted end to end. Tissier et al. [26] proposed an edge possibility prediction approach by using CNN. In their approach, most possible CTU partition results were generated, and the final partition was determined through postprocessing and rate-distortion optimization (RDO). Although methods in this category consider the influence of the entire CTU space information on the splitting of the current CU, they do not take into account the influence of the splitting of parent CUs. Furthermore, for some end-to-end methods, which take a strategy of reducing the CTU partition candidates for RDO, the RDO process is not fully skipped so that the complexity reduction is limited.

To make full use of splitting information from both neighboring and parent CUs, we first propose a new representation for CTU partitioning structure. Specifically, we use an array called split vector (SV) to represent a CTU partitioning structure. Then, we design and train a transformer model to predict the PV of the CTU. With the introduction of the transformer, the CTU partition is regarded as a sequence problem. Finally, the CTU partition structure is decided from SV through postprocessing, and the RDO process is no longer needed for CTU partition searching. The main contributions in this paper are described as follows:

(1) An array called SV is proposed to represent the partition structure of the CTU. With the use of SV, RDO progress searching for optimal CTU partitioning structures can be fully skipped.

(2) We introduce transformer models into the fast determination task for CTU partition structure and imaginatively model the partition problem as a sequence problem affecting each other.

(3) We not only design effective transformer models to predict the SV of the target CTU with high accuracy but also build several datasets for transformer model training.

This paper is organized as follows: Section 1 introduces the background of the proposed approach. Section 2 introduces the fundamental knowledge used in this paper. Section 3 describes the proposed fast algorithm for CTU partitioning. Section 4 shows the experiment results, and Section 5 concludes this paper.

2. Fundamental Knowledge

In this section, we introduce the quadtree structure of HEVC intracoding, and a brief description of the transformer is also given.

2.1. Quadtree Structure of HEVC Intrapartitioning. First, each intraframe is divided into nonoverlapped square blocks called coding tree units (CTUs) which are usually 64×64 pixels. To cope with the texture characteristic of the CTU,

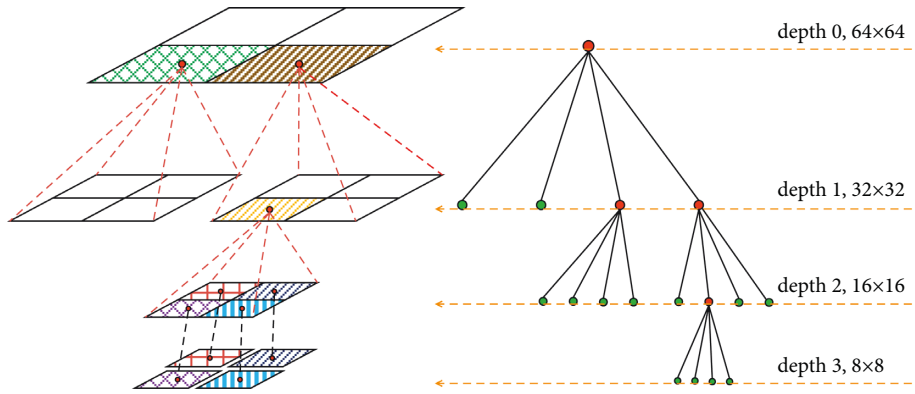

Figure 1: A partition sample of the CTU and the corresponding quadtree structure.

the CTU can be further split into four equal-sized square blocks, and each block can be iteratively further split into four squared sub-blocks according to the quadtree partition structure. In the quadtree of the final CTU partitioning, the CTU serves as the root, and each leaf node represents a coding unit (CU). The CU size of HEVC intracoding varies from 64×64 pixels to 8×8 pixels since the max depth of a quadtree is 3. Figure 1 shows the partitioning result of the CTU and the corresponding quadtree structure. The deeper the quadtree, the smaller the CU.

CTU can be adaptively partitioned into CUs of different sizes to achieve optimal coding efficiency, and the final partition structure is decided according to a brute-force method called rate-distortion optimization (RDO) [27]. RDO evaluates the cost of every possible partition structure in terms of bit rate and visual quality first. As a result, the partition structure with minimal cost is selected as the final partition result for the CTU. Obviously, the encoding time spent on the RDO process increases exponentially with the increase of quadtree depth. Thus, it is essential to reduce the computational burden by replacing the RDO process with an end-to-end approach for CTU partition structure determination.

2.2. Transformer Network. Attention mechanism has been widely used in neural network models such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs) [28]. Since the attention mechanism was proposed, sequence-to-sequence models with attention have shown performance improvement in various tasks. In 2017, Ashish et al. [29] first proposed an attention fully based model named transformer. In order to integrate the advantages of CNNs and RNNs, they creatively used the full attention mechanism to build the network. They applied the transformer to machine translation tasks and achieved state-of-the-art effects at that time.

Like most sequence-to-sequence models, the transformer can be divided into two main parts, i.e., encoder and decoder. The encoder is responsible for mapping the input sequence into a hidden layer, which is the mathematical expression of the input sequence. The decoder then maps the hidden layer back to the target sequence. Using a transformer, we can solve various problems, such as image

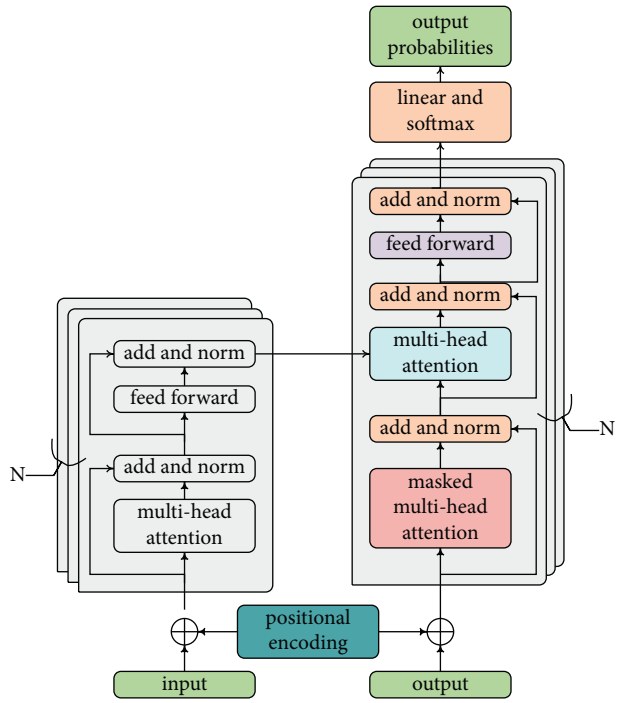

Figure 2: Network structure of the transformer employed by the proposed approach.

classification, summary generation, and machine translation. Figure 2 shows the model structure diagram of the transformer. In Figure 2, the encoder consists of N subencoders, and each subencoder includes two layers, which are a multihead attention mechanism and a fully connected feed-forward network, respectively. Besides, residual connection and normalization are also added to each layer. As we can see from Figure 2, transformer decoders are also composed of N subdecoders, and each subdecoder has one more masked multihead attention layer than the subencoder.

The transformer is a new network architecture designed to replace RNN and CNN. It can be stacked to very deep depth so that it can fully explore the characteristics of a deep neural network and achieve high accuracy. Unlike CNN, which is only able to obtain local information, it can directly obtain global information and capture sequence

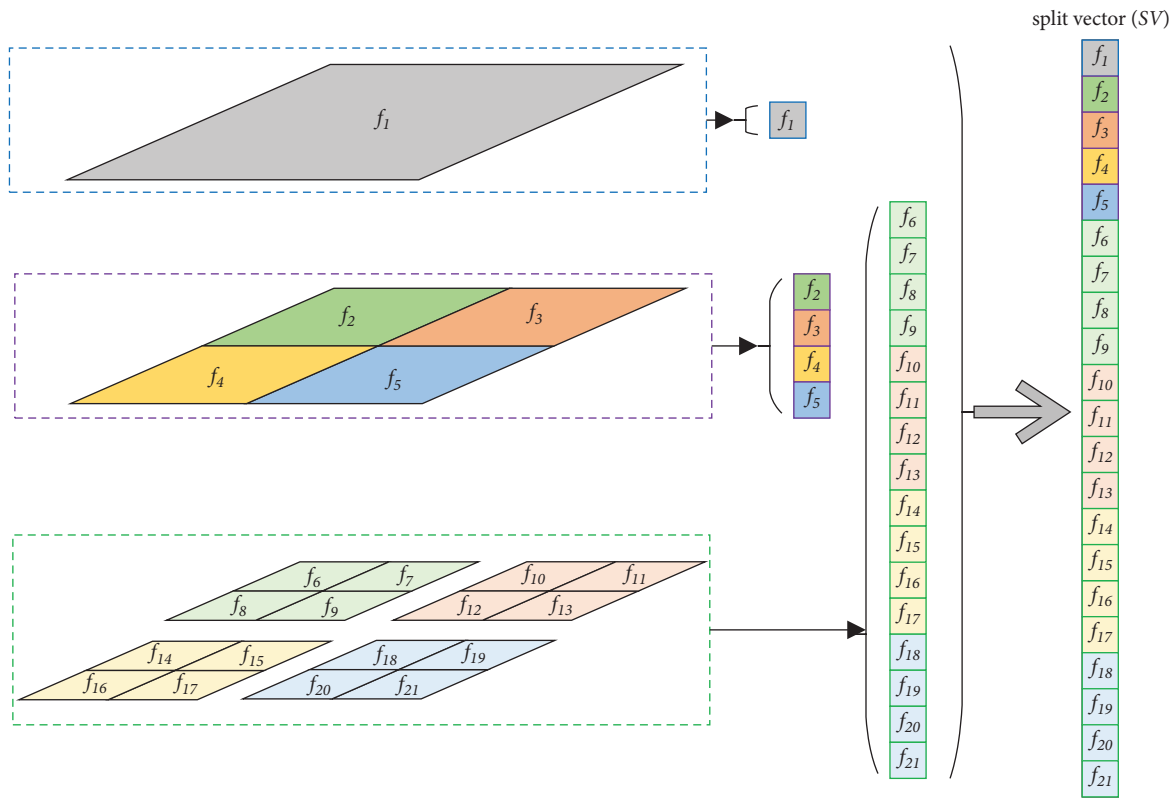

Figure 3: Representation of the CTU partition structure.

dependence. Compared with RNN, the transformer can realize fast parallelism using the attention mechanism, and the training time is greatly reduced.

3. Proposed Method

This section is divided into parts. Firstly, we introduce how the CTU partition structure is represented in the proposed approach. Then, we introduce the training process of transformer models, and loss function designing and train dataset preparation are both included. Finally, we present the postprocessing procedure for the outputs of transformer models.

3.1. Representation of the CTU Partition Structure. In this paper, we propose a novel representation way for the CTU partition structure. Specifically, each of the CTU partition structures can be represented by a vector called split vector (denoted as SV) which contains 21 Boolean split flags. Figure 3 illustrates how the CTU partition structure is reflected in an SV.

According to the quadtree partition rule used in HEVC intracoding, the CTU is split recursively until the max CU depth is reached. Thus, a CTU partition structure can be represented depth wisely. In depth 0, f_1 is the split flag of a CTU. In depth 1, four Boolean values (f_2, f_3, f_4, and f_5 in Figure 3) are used to represent the split status of the four sub-CUs of the current CTU. Furthermore, f_6–f_{21} represent the split flag of each 16×16 CU in depth 2, respectively. Finally, these 21 CU split flags form the SV of the current CTU.

It is convenient and effective to put these flags together into SV. The SV provides us with a new way of reviewing the CTU partitioning jointly, and it avoids treating the CTU partition problem as a level-wise binary classification task by using several cascade prediction models. In other words, SV makes it possible for our method to predict the split flags of blocks in the same CTU jointly due to the high correlation among them.

3.2. Overview of the Proposed Approach. Most existing methods, either statistic-based or learning-based, partition a CTU by deciding whether to split CUs in particular depths. These methods usually employ as many as 3 modes to complete the binary prediction for CUs in different depths. However, with the proposed end-to-end method called TBFA, the partition structure of CTUs is decided through a single prediction by using SV.

Besides, the splitting of the CU is not only highly related to the entire CTU it locates but is also affected by other CUs locating the same CTU. However, many previous methods only focus on the current block without considering information from neighboring or father CUs. BTFA is designed to be able to consider them jointly by using SV, which consists of 21 split flags of neighboring and father CUs in the current CTU.

Also, we specifically design a transformer to predict the SV of the CTU. In BTFA, the transformer is used to explore

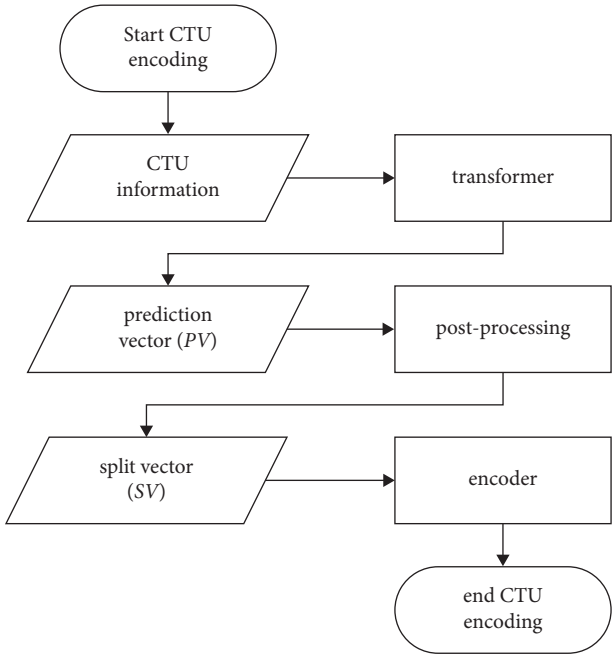

Figure 4: Flowchart of the proposed approach.

the relations among those 21 split flags. The transformer takes the luminance values of the CTU as input and outputs a prediction vector (denoted as PV) of the CTU. Through training the transformer models, PV equals the target SV of the current CTU with high accuracy. Thus, SV can be obtained from PV after the postprocessing procedure. Once the SV is obtained, the entire CTU partition is determined. Then, the encoder can encode each CTU directly, completely avoiding the RDO process for partitioning. Figure 4 shows the whole flow of the proposed TBFA.

3.3. Transformer Training. The transformer predictor plays a key role in BTFA, and a classic transformer structure is employed by our approach. In this part, the design of the loss function is described, and the way in which data are prepared for training is also given.

3.3.1. Loss Function Design. The loss function is used to measure the distance or bias between the ground truth and the predicted result outputted by a model for the current sample. In the case of this paper, the loss function is employed to measure the error between PV and SV.

Actually, PV output by the transformer contains the partition probability of each part corresponding to the current CTU. Conversion from PV output by the transformer to SV is required for encoding, and the conversion process is quite simple that only a comparison between each element and 0.5 is needed. Specifically, if one element in PV is smaller than 0.5, the corresponding element located at the same position in SV is set to 0. On the contrary, if one element in PV is greater than 0.5, the colocated element in SV is set to 1. Contradictory values may exist in SV transformed from PV since the judging of each element of PV is independent during the conversion process. When a CTU is not split, related sub-CUs in depth 1 and 2 are not split either. Thus, if f_1 in Figure 3 equals 0, the values of rest elements in SV are supposed to be 0. To address the inconsistency among values of SV, we carefully designed a loss function for training.

When optimizing the weight parameters of the model iteratively, there is no point in considering the prediction error of the corresponding four sub-CUs if the real label of the current CU is nonsplit. Based on this, we designed the loss function, and the final loss of a sample can be calculated as

$$L = g_1 + l_1 \sum_{i=2}^{5} \left(g_i + l_i \sum_{j=4i-2}^{4i+1} g_j \right), \tag{1}$$

$$g_i = \left((1 - l_i) \log \frac{1}{1 - p_i} + l_i \log \frac{1}{p_i} \right), \tag{2}$$

where l_i denotes the ith element of SV, p_i denotes the ith element of PV, i is an integer from 1 to 21, and g_i is the cross entropy function for each of 21 element pairs.

The loss function used in our model training is designed and improved from cross entropy. It removes the effect brought by contradictory values during the training procedure, improves prediction accuracy, and reduces computational burden as well.

3.3.2. Train Data Preparation. The dataset for training is so important that it can influence model prediction performance greatly. To make sure transformer models achieve their best performance, we construct a reasonable dataset for each model. In this paper, the transformer model is used to make a one-shot prediction of the partition structure for the CTU. Thus, only one transformer model is needed when a video sequence is being encoded. However, the quantization parameter (QP) is one of the key factors affecting CU splitting result, and the partitioning structure of the CTU differs from different QP values. To make our algorithm more specific and achieve higher prediction accuracy and better encoding performance, we train one dedicated transformer model for each QP. To validate the proposed method, we take 4 classic QP values in this paper, and there are a total of 4 datasets constructed for QP 22, 27, 32, and 37, respectively.

We select five videos from standard test sequences and employ HEVC reference software HM16.7 to encode their entire frames under all-intramode and four QP values (22, 27, 32, and 37). These five sequences are *Traffic* (2560 × 1600), *ParkScene* (1920 × 1080), *BasketballDrill* (832 × 480), *BQSquare* (416 × 240), and *FourPeople* (1280 × 720). Sequences used for model training are with different scenes and resolutions, which make trained models more robust and generalized.

Once encoding is completed on these training sequences, we can obtain encoding results of each training sequence under each certain QP. In particular, partitioning structures of all CTUs in training sequences are generated and are further converted to ground truth SVs served as target labels in the training dataset for corresponding QP. As a result,

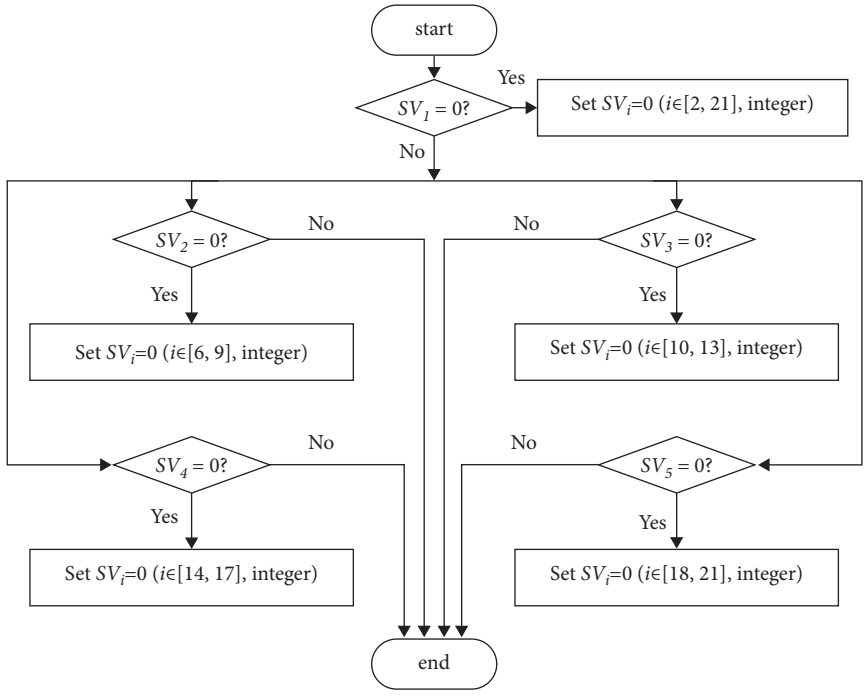

Figure 5: Postprocessing procedure on SV converted from PV.

four training datasets are assembled, and each sample of a dataset consists of luminance pixels and an SV of each CTU. It is worth noting that the only difference among these four datasets is SVs. Samples of different datasets share the same luminance pixel values since they are from the same training sequences.

3.4. Postprocessing of the Split Vector. PV is first output by our transformer model once the derivation is completed. Then, the predicted SV of the current CTU is generated from PV through a conversion process. SV consists of 21 binary elements, and each corresponds to a potential sub-CU of the current CTU. Although the contradiction among SV elements has been mitigated slightly by a well-designed loss function, the influence of loss function designing is not great enough to remove all contradictions in SV. So SV cannot be used directly as the split flags of CUs in the CTU. It is necessary to carry out a postprocessing procedure on SV converted from PV, and the detailed postprocessing procedure is described in Figure 5.

As we can see from Figure 5, the postprocessing procedure is easy enough to be carried out quickly. To be specific, we set SV_2 all the way to SV_{21} to be 0 if SV_1 is 0. If SV_2 is 0, we set SV_6, SV_7, SV_8, and SV_9 to be 0. Similarly, we set SV_{10}, SV_{11}, SV_{12}, and SV_{13} to be 0 if SV_3 is 0. We set SV_{14}, SV_{15}, SV_{16}, and SV_{17} to be 0 if SV_4 is 0. We set SV_{18}, SV_{19}, SV_{20}, and SV_{21} to be 0 if SV_5 is 0. After the fine-tuning described above, SV is able to be used in video encoding finally.

4. Experiment Result

In this section, we first analyze the prediction accuracy of transformer models employed by the proposed approach. Deep learning framework PyTorch is used to complete training and prediction. Simulations were executed on a Windows 7 64 bit operating system workstation with NVIDIA RTX 2080s GPU and Intel(R) Xeon(R) CPU E5-2623 v3 @ 3.00 GHz and 3.00 GHz (2 processors), 64.0 GB.

Then, to verify the effectiveness of the proposed one-shot approach for CTU partitioning, we implemented it on HEVC reference platform HM16.7. Coding parameters, such as additional coding tools, were set as default. Besides, all-intra main configuration was adopted to encode all the frames of video sequences in the standard test set. The BD-rate was employed to evaluate the coding performance of the proposed method, and the time saving ratio denoted by TS was used to measure the complexity reduction of encoding algorithms. It is defined as

$$TS = \frac{\text{time}_o - \text{time}_p}{\text{time}_o}, \tag{3}$$

where time_o denotes time spent by the original HM16.7 encoder and time_p is the time spent by the encoder on which the proposed algorithm is implemented.

At last, partition results of original HM16.7 and the proposed approach are compared on one frame randomly selected. The comparison results visually demonstrate the partitioning effect of the proposed one-shot algorithm for CTU partitioning structure determination.

Table 1: Splitting accuracy of four transformer models at each CU depth.

Depth level	QP 22 (%)	QP 27 (%)	QP 32 (%)	QP 37 (%)
Depth 0	95.58	91.08	91.04	89.38
Depth 1	78.12	81.81	82.44	82.33
Depth 2	74.54	80.60	82.55	84.87

Table 2: Percentage of inferring time to encoding time under different QPs.

	QP 22 (%)	QP 27 (%)	QP 32 (%)	QP 37 (%)
Percentage	0.015	0.021	0.025	0.031

Table 3: Bit rate loss and time saving comparison between the proposed algorithm and state of the art

Class	Sequence	CNNFA		FICUSA		BTFA		Proposed	
		BD-rate (%)	TS (%)	BD-rate (%)	TS (%)	BD-rate (%)	TS (%)	BD-rate (%)	TS (%)
A	*PeopleOnStreet*	3.97	55.59	2.63	45.00	1.16	38.06	4.03	59.04
	Average	**3.97**	**55.59**	**2.63**	**45.00**	**15.41**	**38.06**	**4.03**	**59.04**
B	*Kimono*	2.38	72.72	0.42	16.27	3.32	59.23	6.50	75.35
	Cactus	6.02	62.98	2.44	37.61	1.11	44.95	4.70	66.02
	BasketballDrive	6.02	69.51	1.46	27.96	2.46	49.09	5.43	71.68
	BQTerrace	4.82	57.89	2.41	41.10	1.09	46.88	2.66	58.07
	Average	**4.81**	**65.78**	**1.68**	**30.74**	**2.00**	**50.04**	**4.82**	**67.78**
C	*BQMall*	8.08	52.14	4.33	44.72	0.97	41.52	4.44	56.98
	PartyScene	9.45	58.75	4.66	46.06	1.36	37.68	1.49	44.07
	RaceHorses	4.42	58.19	—	—	0.60	47.49	2.70	58.22
	Average	**7.32**	**56.36**	**4.50**	**45.39**	**0.98**	**42.23**	**2.88**	**53.09**
D	*BasketballPass*	8.40	64.02	3.34	41.48	0.67	36.34	2.32	48.14
	BlowingBubbles	8.33	60.78	3.09	43.45	0.14	25.95	0.92	33.65
	RaceHorses	4.95	57.29	—	—	0.73	33.76	1.88	42.56
	Average	**7.23**	**60.70**	**3.22**	**42.47**	**0.51**	**32.02**	**1.71**	**41.45**
E	*Johnny*	7.96	66.55	2.23	38.56	2.38	63.13	7.45	73.12
	KristenAndSara	5.48	64.72	3.07	43.19	2.09	58.57	6.19	71.87
	Average	**6.72**	**65.64**	**2.65**	**40.88**	**2.24**	**60.85**	**6.82**	**72.49**
Overall average		**6.01**	**60.81**	**2.93**	**40.89**	**4.23**	**44.64**	**3.90**	**58.77**

4.1. Splitting Accuracy. Accuracy of splitting decisions predicted by transformer models for CUs of each depth is shown in Table 1. Depth levels in Table 1 represent different sizes of the CU, accuracy of depth level 0 means the percentage of right splitting decisions predicted by transformer model for CTUs, and accuracy of depth levels 1 and 2 means the percentage of right splitting decisions for CUs of size 32×32 and 16×16, respectively.

As we can see from Table 1, every transformer model achieves its own highest accuracy on the splitting prediction on depth level 0, and accuracies on levels 1 and 2 do not have much difference and are lower than that on level 0 by around 10%. The reason is that splitting prediction of large CUs is easier than that of small CUs since large CU provides more information for prediction, and texture is more obvious. Another phenomenon is that prediction accuracy of transformer models at depth level 0 decreases as QP increases, while the accuracy at depth level 1 and 2 increases as QP increases. It means that the transformer employed by the proposed approach is good at predicting the splitting results of large CUs when QP is small, while it predicts the splitting results of small CUs easily under large QP values. These properties provide guidance for the industry application of our method.

4.2. Inferring Time Overhead. The proposed approach aims at reducing the encoding time of HEVC by employing transformer models. However, inferring time of transformers must be considered due to their parameter scale. During the encoding process, the inferring time is included for a fair performance evaluation. Besides, we calculate the percentage of inferring time to encoding time under different QPs, and the results are shown in Table 2. As we can see from Table 2, the inferring time overhead of transformers takes a quite low percentage of the encoding time while the proposed method is used, which is because we use GPUs to complete the inferring calculation of transformers.

4.3. Encoding Performance. The last two columns in Table 3 show the encoding performance of the proposed method. Compared to original HM16.7, 58.77% encoding time on

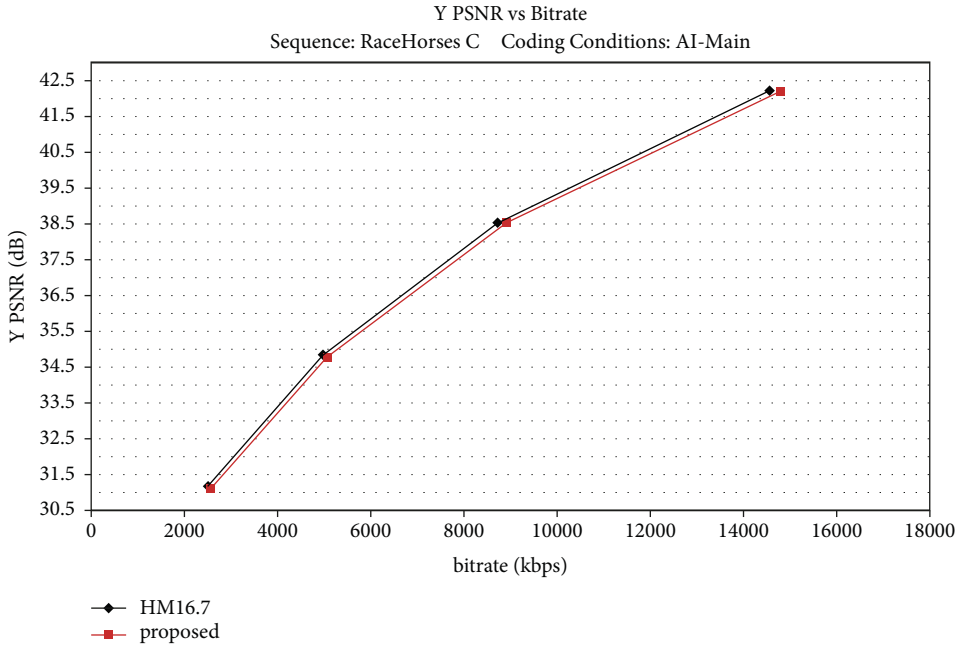

Figure 6: Rate-distortion curve comparison of the proposed approach and original HM16.7 on sequence *RaceHorses* (832 × 480).

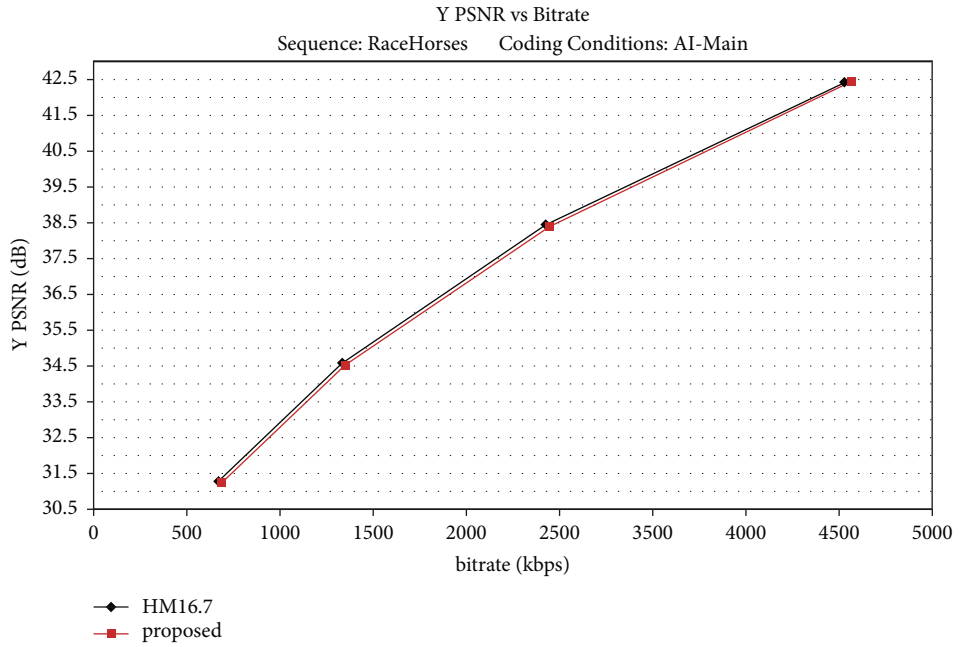

Figure 7: Rate-distortion curve comparison of the proposed approach and original HM16.7 on sequence *RaceHorses* (416 × 240).

average is saved while sacrificing about 3.90% bit-distortion rate. Besides, we observe that our fast approach achieves better results among sequences of resolution class *C* and *D* in terms of BD-rate loss, and time saving is acceptable. Though the proposed algorithm is able to save as much as 59.04%, 67.78%, and 72.49% encoding time on resolution class *A*, *B*, and *E*, respectively, BD-rate loss is not quite low while acceptable. It means that our transformer-based one-shot approach works better on the sequence of low resolution. Especially, we find encoding results to be good and balanced on sequences *RaceHorses* (832 × 480) and *RaceHorses* (416 × 240), and their rate-distortion (RD) curves are shown in Figures 6 and 7, respectively. As we can see from Figures 6 and 7, though the proposed approach saves much encoding time on these two sequences, their RD curves are both quite close to that of original HM16.7, which means the visual difference is negligible for the proposed approach. Observing contents of these two sequences, we find that they

Table 4: Encoding performance of the proposed approach on different resolution classes.

Class	ESDA		BTRNFA		Proposed	
	BD-rate (%)	TS (%)	BD-rate (%)	TS (%)	BD-rate (%)	TS (%)
A	1.00	48.50	2.28	62.21	4.03	59.04
B	1.16	45.60	2.69	70.66	4.82	67.78
C	0.55	33.75	1.32	52.82	2.88	53.09
D	0.30	33.00	0.88	41.86	1.71	41.45
E	2.73	61.33	3.30	74.19	6.82	72.49
Average	**1.15**	**44.44**	**2.09**	**60.35**	**3.90**	**58.77**

Figure 8: Partition result comparison between the proposed approach and original HM16.7.

both contain objects moving fast, which indicates our fast approach is ideal for industrial applications working in scenes involving a lot of movement.

To further analyze the coding performance of the proposed algorithm, we compare it with three recent studies. Respectively, they are convolution neural network-based fast algorithm (CNNFA) proposed by Liu et al. [18], fast intra-CU splitting algorithm (FICUSA) proposed by Zhong et al. [30], and bagged tree-based fast algorithm (BTFA) proposed by Li et al. [31]. CNNFA, FICUSA, and BTFA are all effective schemes for intra-CU size decision of HEVC, and their performance on standard test sequences is listed in Table 3. As we can see, FICUSA provides 0.97% less BD loss compared to ours, but the time saving of our method outperforms about 18%. Considering CNNFA, though its time saving is slightly higher than ours by 2.04%, their BD loss is also higher by 2.11%. Moreover, compared with BTFA, our approach outperforms in terms of both BD-rate and time saving.

Moreover, we also compared the proposed approach with other two fast partition approaches, which are the effective CU size decision approach (ESDA) [22], bagged tree, and ResNet joint fast approach (BTRNFA) [32]. Table 4 shows the encoding performance of ESDA and BTRNFA on different sequence resolution classes. As we can see from Table 4, the proposed approach outperforms ESDA in terms of time saving, while it causes more BD-rate loss. Compared with BTRNFA, the proposed approach makes more BD-rate loss by 2.85% and almost the same time saving. Though the proposed approach does not defeat BTRNFA, it does not require feature extraction and finishes partition prediction using as few as one transformer model.

Tables 3 and 4 prove that the performance of the proposed transformer-based fast approach for CTU partitioning is satisfactory and competitive and has good capacity in practical industrial applications according to the comprehensive performance of various coding scenarios.

4.4. Partition Comparison. To visualize the encoding performance of the proposed fast approach for HEVC intra-coding, we give the partition results predicted by our algorithm on the 200th frame of sequence *Basketball Pass* (416×240) under QP 22. Black, red, and green lines in Figure 8 represent borders of CUs for final encoding. To verify the correctness of CU splitting, we compare the partitioning results of our approach and original HM16.7, and differences are shown with red and green lines in Figure 8. The black line represents that our algorithm and original HM16.7 share the same partition results. The green line represents the boundaries of CUs, which are split by original HM16.7 but are not split by our approach.

Boundaries of CUs decided nonsplit by original HM16.7 but are split by our approach are shown with red lines in Figure 8.

As we can see from Figure 8, the partition results of the proposed algorithm are consistent with those of original HM16.7 in most CU cases. Splitting results of CUs located on main subjects in a frame are almost the same, while differences mainly exist in the background region of a frame. Compared with original HM16.7, the proposed algorithm is more likely to split the CU. Red lines outnumber green lines, which indicates that the splitting errors are far more than nonsplitting errors of the proposed approach.

5. Conclusion

In this paper, we focus on speeding up the video encoding process of industry applications. As a result, we propose a transformer-based fast CTU partitioning algorithm for HEVC intracoding. We convert the CTU partitioning structure to a split vector and employ transformer models to predict that of the target CTU while encoding. Compared with the original HM 16.7 encoder, our approach reduces encoding time by 58.77% on average while sacrificing negligible rate-distortion performance on selected video sequences. Intensive analysis and experiments show that the proposed solution has great capacity for working in industry applications, especially for scenes involving a lot of movement.

Data Availability

Data used and analyzed during the current study are available from the corresponding author on reasonable request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Key Research and Development Program of China (Grant no. 2020YFB1805403), the National Natural Science Foundation of China (Grant no. 62032002), the Natural Science Foundation of Beijing Municipality (Grant no. M21034), and the 111 Project (Grant no. B21049).

References

[1] P. Hořejší, K. Novikov, and M. Šimon, "A smart factory in a smart city: virtual and augmented reality in a smart assembly line," *IEEE Access*, vol. 8, pp. 94330–94340, 2020.

[2] L. Wang and Z. Zhang, "Automatic detection of wind turbine blade surface cracks based on UAV-taken images," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 9, pp. 7293–7303, 2017.

[3] H. Yang, C. Huang, L. Wang, and X. Luo, "An improved encoder–decoder network for ore image segmentation," *IEEE Sensors Journal*, vol. 21, no. 10, pp. 11469–11475, 2021.

[4] M. Wang, W. Xie, J. Zhang, and J. Qin, "Industrial applications of UHD video coding with an optimized super-SAO framework," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 12, pp. 7613–7623, 2020.

[5] N. Dolati, A. Beheshti, and H. Azadegan, "A selective encryption for H.264/AVC videos based on scrambling," *Multimedia Tools and Applications*, vol. 80, no. 2, pp. 2319–2338, 2021.

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

[7] W. Wang, J. Le, Z. Wang et al., "Event-triggered consensus control for high-speed train with time-varying actuator fault," *IEEE Access*, vol. 8, pp. 50553–50564, 2020.

[8] C. Huang, L. Wang, X. Luo, H. Zhang, and Y. Song, "Evolutionary computing assisted deep reinforcement learning for multi-objective integrated energy system management," in *Proceedings of the IEEE 33rd International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 506–511, IEEE, Washington, DC, USA, November 2021.

[9] Z. Wang, L. Wang, C. Huang, Z. Zhang, and X. Luo, "Soil-moisture-sensor-based automated soil water content cycle classification with a hybrid symbolic aggregate approximation algorithm," *IEEE Internet of Things Journal*, vol. 8, no. 18, pp. 14003–14012, 2021.

[10] C. Huang, H. Zhang, Y. Song, L. Wang, T. Ahmad, and X. Luo, "Demand response for industrial micro-grid considering photovoltaic power uncertainty and battery operational cost," *IEEE Transactions on Smart Grid*, vol. 12, no. 4, pp. 3043–3055, 2021.

[11] L. Li, K. Ota, and M. Dong, "Deep learning for smart industry: efficient manufacture inspection system with fog computing," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4665–4673, 2018.

[12] T. Cerquitelli, D. J. Pagliari, A. Calimera et al., "Manufacturing as a data-driven practice: methodologies, technologies, and tools," *Proceedings of the IEEE*, vol. 109, no. 4, pp. 399–422, 2021.

[13] F. Al-Turjman and B. D. Deebak, "Seamless authentication: for IoT-big data technologies in smart industrial application systems," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 4, pp. 2919–2927, 2020.

[14] M.-W. Tian, A. Mohammadzadeh, J. Tavoosi et al., "A deep-learned type-3 fuzzy system and its application in modeling problems," *ActaPolytechnicaHungarica*, vol. 19, no. 2, pp. 151–172, 2022.

[15] X. Dong, L. Shen, M. Yu, and H. Yang, "Fast intra mode decision algorithm for versatile video coding," *IEEE Transactions on Multimedia*, vol. 24, no. 2021, pp. 400–414, 2022.

[16] C. Ma, A. Mohammadzadeh, H. Turabieh, M. Mafarja, S. S. Band, and A. Mosavi, "Optimal type-3 fuzzy system for solving singular multi-pantograph equations," *IEEE Access*, vol. 8, pp. 225692–225702, 2020.

[17] F. Zaki, A. E. Mohamed, and S. G. Sayed, "CtuNet: a deep learning-based framework for fast CTU partitioning of H265/HEVC intra- coding," *Ain Shams Engineering Journal*, vol. 2021, no. 1, 2021.

[18] 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.

[19] 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.

[20] K. Kim and W. W. Ro, "Fast CU depth decision for HEVC using neural networks," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 5, pp. 1462–1473, 2019.

[21] J. Shi, C. Gao, and Z. Chen, "Asymmetric-kernel CNN based fast CTU partition for HEVC intra coding," in *Proceedings of the 2019 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2019.

[22] L. Shen, Z. Zhang, and Z. Liu, "Effective CU size decision for HEVC intracoding," *IEEE Transactions on Image Processing*, vol. 23, no. 10, pp. 4232–4241, 2014.

[23] L. Shen, Z. Zhang, and Z. Liu, "Adaptive inter-mode decision for HEVC jointly utilizing inter-level and spatiotemporal correlations," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 10, pp. 1709–1722, 2014.

[24] M. J. Chen, C. A Lee, Y. H. Tsai et al., "Efficient partition decision based on visual perception and machine learning for H. 266/Versatile video coding," *IEEE Access*, vol. 10, no. 2022, pp. 42141–42150, 2022.

[25] A. Feng, C. Gao, L. Li, and L. Dong, "Cnn-based depth map prediction for fast block partitioning in HEVC intra coding," in *Proceedings of the 2021 IEEE International Conference on Multimedia and Expo (ICME)*, July 2021.

[26] A. Tissier, W. Hamidouche, J. Vanne, and F. Galpin, "CNN oriented complexity reduction of VVC intra encoder," in *Proceedings of the 2020 IEEE International Conference on Image Processing (ICIP)*, 25-28 October 2020.

[27] B. Huang, Z. Chen, and K. Su, "Low-complexity rate-distortion optimization for HEVC encoders," *IEEE Transactions on Broadcasting*, vol. 2021, no. 99, pp. 1–15, 2021.

[28] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *Computer Science*, vol. 2014, 2014.

[29] V. Ashish, B. Samy, and B. Eugene, "Tensor2Tensor for neural machine translation," *Machine Learning*, 2018.

[30] W. Zhong, C. Dong, and X. Yao, "Fast intra-coding unit splitting algorithm based on spatial-temporal correlation in HEVC," *Journal of Image and Graphics*, 2018.

[31] Y. Li, L. Li, Y. Fang, H. Peng, and Y. Yang, "Bagged tree based frame-wise beforehand prediction approach for HEVC intra-coding unit partitioning," *Electronics*, vol. 9, no. 9, p. 1523, 2020.

[32] Y. Li, L. Li, Y. Fang, H. Peng, and N. Ling, "Bagged tree and ResNet-based joint end-to-end fast CTU partition decision algorithm for video intra coding," *Electronics*, vol. 11, no. 8, p. 1264, 2022.