*electronics*

MDPI

*Article*

# A Complexity Reduction Method for VVC Intra Prediction Based on Statistical Analysis and SAE-CNN

**Jinchao Zhao, Pu Dai and Qiuwen Zhang ***

College of Computer and Communication Engineering, Zhengzhou University of Light Industry, Zhengzhou 450002, China; 2004086@zzuli.edu.cn (J.Z.); 332007010490@zzuli.edu.cn (P.D.)
* Correspondence: 2012032@zzuli.edu.cn; Tel.: +86-371-8660-9559

**Abstract:** Compared with High Efficiency Video Coding (HEVC), the latest video coding standard Versatile Video Coding Standard (VVC), due to the introduction of many novel technologies and the introduction of the Quad-tree with nested Multi-type Tree (QTMT) division scheme in the block division method, the coding quality has been greatly improved. Due to the introduction of the QTMT scheme, the encoder needs to perform rate–distortion optimization for each division mode during Coding Unit (CU) division, so as to select the best division mode, which also leads to an increase in coding time and coding complexity. Therefore, we propose a VVC intra prediction complexity reduction algorithm based on statistical theory and the Size-adaptive Convolutional Neural Network (SAE-CNN). The algorithm combines the establishment of a pre-decision dictionary based on statistical theory and a Convolutional Neural Network (CNN) model based on adaptively adjusting the size of the pooling layer to form an adaptive CU size division decision process. The algorithm can make a decision on whether to divide CUs of different sizes, thereby avoiding unnecessary Rate–distortion Optimization (RDO) and reducing coding time. Experimental results show that compared with the original algorithm, our suggested algorithm can save 35.60% of the coding time and only increases the Bjøntegaard Delta Bit Rate (BD-BR) by 0.91%.

**Keywords:** VVC; SAE-CNN; complexity reduction; CU division decision



**Citation:** Zhao, J.; Dai, P.; Zhang, Q. A Complexity Reduction Method for VVC Intra Prediction Based on Statistical Analysis and SAE-CNN. *Electronics* **2021**, *10*, 3112. https://doi.org/10.3390/electronics10243112

Academic Editor: Cataldo Guaragnella

Received: 10 November 2021
Accepted: 9 December 2021
Published: 14 December 2021



## 1. Introduction

In the information age, video, as an important carrier of information, is integrated into people's lives through computers, TVs, mobile phones and other devices, and has become an indispensable part of today's society. However, with the continuous improvement of video resolution and frame rate, the amount of data occupied by a video is also increasing. Taking a 40-min high-definition color video as an example, using the mainstream 8-bit pixel depth and 30 frames per second playback speed to calculate, its data volume is $1920 \times 1080 \times 3 \times 8 \times 30 \times 60 \times 40 = 3{,}583{,}180{,}800{,}000$ bits, which is obviously not conducive to video storage and real-time transmission. In addition, in the process of video transmission, the reconstructed image will be distorted due to the influence of channel noise [1,2]. In order to realize the compression processing of the video, to facilitate the transmission and application, the video coding technology comes into being.

In order to achieve uniformity in video coding technology and enable video transmission on different platforms, the International Telecommunication Union's Telecommunication Standardization Sector (ITU-T) formulated the earliest video coding standard H.261, followed by H.263 and H.264. The emergence of these standards has greatly promoted the development and application of video technology. The current mainstream of video coding standard is the High Efficiency Video Coding (HEVC), which was formulated by the Video Coding Experts Group (VCEG) and was confirmed as an international standard in 2013. The key technologies were introduced in HEVC, including the quad-tree structure of Coding Unit (CU) partition, the expansion of prediction block size, the intra-frame prediction based on up to 33 directions, the multi-frame motion compensation prediction, etc.

These greatly improve the coding performance [3]. With the rise of new applications such as Ultra-high Definition (UHD), 4K and 3D video, better visual quality and more realistic viewing experience can be provided, making them suitable for Image Maximum (IMAX) movies, video calls, TV broadcasts, high-definition video surveillance and so on, including applications in medical imaging [4]. However, the existing coding standard HEVC can no longer meet the increasing needs of users. In 2015, the Joint Video Exploration Team (JVET) explored the next generation of video coding standards based on HEVC and named it Versatile Video Coding (VVC) [5]. VVC improves the coding technology on the basis of HEVC and proposes some novel technologies, such as: the Quad-tree with nested Multi-type Tree (QTMT) structure of the CU partition, supporting 65 kinds of intra prediction directions, the Multi-reference Line (MRL) intra prediction, the Adaptive Multiple Core Transform (AMT) and so on. These new technologies enable the latest version of the VVC reference software VTM10.0 to increase the coding efficiency by 24% compared with the HEVC reference software HM16.20, but the coding time has increased by 27 times. The main reason is that a more flexible QTMT structure of block partition was adopted in VVC; that is, the horizontal binary-tree, vertical binary-tree, horizontal trinomial-tree and vertical trinomial-tree structure were added on the basis of the quad-tree structure, as shown in Figure 1. Due to the characteristics of the QTMT structure, the encoder can make more flexible divisions for CUs with different texture complexities, thereby improving coding performance [6].

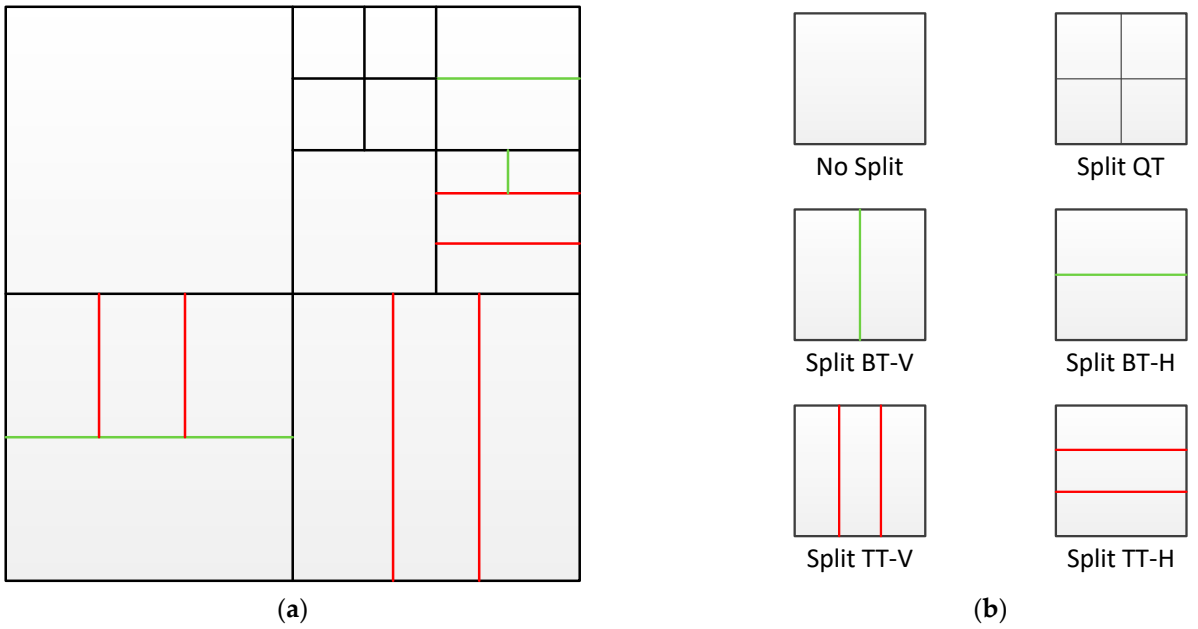



**Figure 1.** Coding Tree Unit (CTU) partitioning in VVC: (**a**) Take a 64 × 64 CU as an example; (**b**) VVC split types.

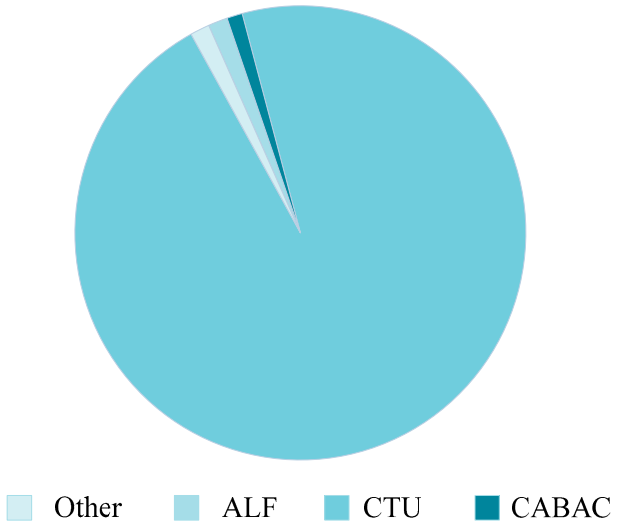
The division process is as follows: For a 128 × 128 Coding Tree Unit (CTU), firstly, the CTU is divided into a quad-tree to obtain four 64 × 64 CUs; secondly, for the 64 × 64 CU, there are two options: quad-tree split and non-split; next, to divide the 32 × 32 CU, there are four options: binary-tree, trinomial-tree, quad-tree split and non-split; and so on, until it is divided into a 4 × 4 CU. It should be noted that the basis used to determine the division of the CU is the Rate–distortion Optimization (RDO). That is, it traverses all possible division modes of the CU, calculates the Rate–distortion (RD) cost and selects the division mode with the smallest rate–distortion as the best division mode of the current CU [7]. The calculation formula of the RD cost J is as follows:

$$J_{\text{mode}} = D + (\lambda \times R_{\text{mode}}), \quad (1)$$

$$D = SSE_{luma} + (W_{chroma} \times SSE_{chroma}), \quad (2)$$

where $R_{\text{mode}}$ represents the encoding bit in the homologous mode; $SSE_{chroma}$ indicates the sum of chromaticity squares of the initial picture and the re-established picture; Similarly, $SSE_{luma}$ represents the sum of luminance squares; $\lambda$ and $W_{chroma}$ are the Lagrange multiplier and chromaticity distortion weights, respectively.

As shown in Figure 2, we use the VTM.10.0 version of the test software to encode the 50-frame (Quantization Parameter, QP = 22) *Basketballdrilltext* sequence under the All Intra (AI) configuration and count the proportion of encoding time occupied by each crucial technique of VVC. Among them, the CTU division time accounts for 96.78% of the overall coding time, and the summation of the time used by remaining technologies just accounts for 3.22%. It can be seen that facile CU division makes a significant contribution to the improvement of VVC encoding performance, but this technology also occupies the largest computational complexity of VVC.




(**a**)

| Key technology | Coding time | Proportion |
|---|---|---|
| ALF | 77.1 s | 1.34 % |
| CTU | 5568.3 s | 96.78 % |
| CABAC | 43.8 s | 0.76 % |
| Other | 64.4 s | 1.12 % |
| Total coding time | 5753.6 s | 100 % |

(**b**)

**Figure 2.** Chart of encoding complexity: (**a**) Scale chart of encoding time occupied by various key technologies; (**b**) scale table of encoding time occupied by various key technologies.

Through the previous analysis of the VVC technology and the experiments on the coding time it took, we found that a fast CU partitioning method that reduces the unnecessary RDO for some CUs is of great significance for reducing the coding time and complexity of VVC. Therefore, this paper proposes a VVC intra prediction complexity reduction algorithm based on statistics and the Size-adaptive Convolutional Neural Network (SAE-CNN).

The remaining chapters are arranged as follows: Section 2 will introduce some preparations for the paper, including an analysis of the algorithms proposed by the predecessors and the motivation of the algorithm. Section 3 will introduce the proposed algorithm, including the overall process of the proposed algorithm, CU partition pre-decision algorithm, the framework of SAE-CNN proposed in this paper, the establishment of the data set and the training program. Section 4 will exhibit the experimental outcomes of the algorithm proposed in this paper and compare and analyze the original algorithm and other algorithms. Lastly, Section 5 will give the conclusion of the whole article.

## 2. Related Works

In modern years, artificial intelligence algorithms have developed rapidly and are gradually applied to various fields [8–10], including video coding. In the past, many

artificial intelligence-related methods have been suggested to speed up the CU division process in video coding.

### *2.1. Methods for Former Standards*

Before VVC, the mainstream of video coding standards included HEVC, AVS2 and VP9. Among them, HEVC has become an internationally recognized coding standard and has been extensively studied. To speed up the block partitioning process and reduce coding complexity, some artificial intelligence algorithms, such as deep learning, machine learning and reinforcement learning, are widely used in video coding. Heindel et al. [11] proposed SVM-based fast block split decisions for HEVC inter coding based on Zhang et al. [12], using the Support Vector Machine (SVM) classifier to select skip flag, coded unit flag, RD cost, initial bits and average neighboring depth as features. The CU is directly partitioned and decided without calculating the RD cost. In the case that SVM cannot make accurate decisions, RDO is used as a backup plan. Xu et al. [13] recommended a fast CTU depth selection algorithm based on machine learning. First, the CTU is initially divided into depth prediction based on QP and texture complexity, and then rate and distortion are used as feature input, and the NN classifier is used to convert the block partition problem as modeled into a two-class classification problem. The depth of the CU is selected, and the RD cost calculation process of the depth prediction of some coding units is skipped, which saves coding time. Jamali et al. [14] suggested a block partition decision algorithm based on deep reinforcement learning (RL). The CU size decision was regarded as a sequence decision problem and expressed through reinforcement learning, using A to find the best coding strategy for CU size decision in an intra coding environment. Kim et al. [15] recommended a fast-coding unit depth decision algorithm based on the Convolutional Neural Network (CNN), which uses a CNN to predict CTU depth instead of an exhaustive search to calculate the RD cost. For each depth judgment, a different CNN model is used to make predictions. The proposed CNN model is suitable for both intra- and inter-prediction, and vector data is added to the fully connected layer to improve the prediction accuracy. Zhang et al. [16] recommended a CNN-based CU partition decision algorithm. A $64 \times 64$ CU was input to CNN, and 21 partition flags were obtained, and the CU was partitioned directly according to the flag information. Guo et al. [17] suggested a lightweight CNN-based intra-coding complexity reduction algorithm. The output layer of CNN is composed of three branches, which represent the division of CUs with dimensions of $64 \times 64$, $32 \times 32$, and $16 \times 16$. Through this CNN, it is possible to predict the overall division result of a $64 \times 64$ CTU with one attempt. The above artificial intelligence algorithms for HEVC fast block division show good performances in reducing coding time. However, since VVC has a significant difference in the division structure compared with HEVC, the above algorithms cannot be well applied to VVC.

### *2.2. Approaches for VVC*

Because VVC is in the process of continuous improvement, only a small group of people have proposed some artificial intelligence algorithms to apply VVC fast block division methods, most of which are based on neural networks or based on machine learning algorithms instead of RDO to directly divide CU operations or make partitioning decisions to reduce coding complexity. Fu et al. [18] proposed a fast CU division method based on Bayesian decision rules. The algorithm takes the partition type and intra prediction mode of the coding unit as the feature input, makes full use of the binary partition information and designs an early termination algorithm to avoid unnecessary RDO exploration for some CUs. Cui et al. [19] suggested a block partitioning decision algorithm based on directional gradients. By analyzing the division characteristics and direction gradients of CU, the corresponding CU partitioning rules were found to replace the traditional RDO and reduce the complexity. Wu et al. [20] recommended a fast block partitioning method based on SVM, which predicts the type of coding unit partitioning through texture information and designs different classifiers for CUs of different sizes to improve the prediction accuracy.

Tissier et al. [21] suggested a CNN-based intra-encoder complexity reduction algorithm on the basis of Galpin et al. [22], designed a classic Residual Network (ResNet) CNN, input a 64 × 64 CU and output 480 probabilities. For the composed probability vector, each probability corresponds to whether each 4 × 4 boundary is divided, and then each predicted probability is compared with the artificially set threshold. If it is greater than the threshold, it will be divided, otherwise it will not be divided. Li et al. [23] designed a CNN with an early exit mechanism. The CTU division process is divided into multiple stages. Each stage is composed of a conditional convolutional neural network and sub-networks, and then the threshold and the division of the CNN output probability are compared to determine the current CU division mode. Wang et al. [24] designed a fast CNN-based on Quad-Tree Binary-Tree (QTBT) partitioning decision algorithm for inter coding through the statistical analysis of QTBT to guide the design of the CNN architecture. In addition, to achieve very good performance, a motion-compensated residual block was used as the input and the correlation between the reference frame and the current frame was considered. Tang et al. [25] designed a block partitioning decision-making scheme based on the multi-variable pooling layer CNN. For a residual block, the Soble operator is first used to calculate the average gradient value of the CU, then sets the threshold condition to filter out the CUs for which the division decision is difficult to judge, and finally uses CNN to make the division decision on the CU. However, the CNN proposed in this algorithm is too simple to fully extract the feature information of the CU, resulting in unsatisfactory final prediction results. In this article, we borrow this idea to rebuild the CNN model to make up for this deficiency, which will be discussed in detail in the following section. Hang et al. [26] proposed an intra-frame coding acceleration algorithm based on ResNet and a random forest classifier. For a 32 × 32 brightness block, the algorithm is first processed by CNN to obtain the range of division depth, and then a Random Forest Classifier (RFC) is used to obtain the division type of the CU. Fu et al. [27] suggested a block division decision-making scheme based on multi-branch CNN, which is divided into two stages. In the first stage, a 32 × 32 CU is sent to the CNN as an input, and the output result includes the type of CU, the depth of Quad-Tree (QT) division and whether to use TT division. In the second stage, the RDO depth range exploration table is built, and according to the output results of CNN, obtaining the depth and classification exploration range of RDO is carried out, so as to reduce the computational complexity. The above methods are all designed to decrease the computational complexity of the VVC, which greatly shortens the encoding time, but also causes a substantial increase in the bit rate and does not balance the relationship between the encoding complexity and the encoding quality well.

### *2.3. Motivation*

The original block can embody the characteristics of the block to a certain degree, so the current method carried out for VVC using CNN for fast block partition is to send the original block as an input to the CNN for processing. Compared with the original block, the residual block more closely reflects the texture details of the image and the CU division trend. Therefore, if the residual block of a coding unit is sent to the CNN as an input to make a division decision, a more accurate prediction effect will be obtained, and the prediction accuracy will be improved. Therefore, this paper proposes replacing the original block with the residual block as the input of CNN to further improve the accuracy of the division decision.

At the same time, this paper proposes a pre-decision-making scheme for CU partition. According to the correlation among texture complexity, quantization parameters and CU division, a pre-decision dictionary is constructed, and some CUs with obvious division trends are selected, so that they can directly proceed to the subsequent division mode selection without using CNN; in the same way, some CUs with obvious non-division trends can be selected, and there is no need to make division decisions for them, and the entire division process is ended in advance, thereby reducing the coding time to a certain extent.

In addition, because the QTMT division structure is quoted in VVC, CTU can choose a variety of division methods, as the CU size obtained after division is no longer only square as in HEVC, but rectangles of different sizes, such as 32 × 16, 16 × 8, 8 × 4 and so on. If different CNN models are designed for different sizes of coding units, they will not only bring a lot of unnecessary work, but also reduce the utilization rate of the neural network and cause a waste of resources. Therefore, this paper proposes a SAE-CNN, which adaptively adjusts the size of the pooling layer according to the size of the input CU residual block, so that CNN can make decisions on whether to divide CUs of different sizes. Additionally, it replaces part of RDO, which is of great significance for improving the utilization of neural networks and reducing coding complexity.

## 3. Proposed Method

In recent years, CNN has been applied in many fields as an emerging artificial intelligence algorithm, including video coding. In this section, we design SAE-CNN to determine whether the CU is divided in intra-frame prediction and build a division decision plan that can adapt to the size of the CU. The entire algorithm flow is shown in Figure 3. The method designed in this article is aimed at the luminance CU. First, based on statistical analysis, we establish a pre-decision dictionary. For a CU, first the pre-decision judgments based on the pre-decision dictionary are made based on whether it is square or rectangular. **For a CU, whether it is square or rectangular, first make pre-decision judgments based on the pre-decision dictionary.** There are three situations (split, not split and uncertain). The CU residual block is only sent to SAE-CNN when the result is uncertain, and it is judged whether a division operation on the CU should be performed according to the output result. If the result is 'split', then the subsequent division mode selection is performed to select a suitable division mode, and the division process is ended; if the result is 'not split', the division process is directly ended. The following mainly introduces the pre-decision process, SAE-CNN construction and training process.

### *3.1. Pre-Decision Algorithm*

Generally, the decision as to whether a CU will be divided is inseparable from the content and texture it contains. In detail, a CU with a more complex texture is more likely to be divided, and a simpler CU is less likely to be divided. In the same way, the QP selected in the encoding process also affects the judgment of CU division. Therefore, this paper constructs a pre-decision dictionary based on the texture complexity and the correlation between QP and the CU division and uses the dictionary to enact pre-division and judge the CU. The CUs with obvious division trends and obvious non-division trends are selected, and the division operation is directly performed, or the division process of the CU is terminated, thereby avoiding unnecessary RD cost calculation and the CNN prediction process. The construction process of the pre-decision dictionary is as follows:

Select VVC standard test sequences (*Bqterrac*, *Basketballdrill*, *Bqsquare*, *Fourpeople*) with different texture complexity, different scenes and different resolutions. After VTM-10.0 uses different QPs for encoding, it counts the division results of CUs of various sizes. Taking QP as the X axis (QP range is 0~51, and it is quantified in 1 units) and the entropy value of CU residual block as the Y axis (The entropy value in the test sequence ranges from 0.5 to 7.5, and it is quantified in 0.1 units), the pre-decision dictionary is established as shown in Figure 4. Among them, "×" indicates that under the current QP and entropy conditions, the number of CUs divided is much greater than the number of CUs not divided, and it is marked as "split". "×" indicates that under the current QP and entropy conditions, the number of CUs divided is much smaller than the number of CUs not divided, and it is marked as "not split". "×" indicates that under the current QP and entropy conditions, the number of CUs divided is not much different from the number of CUs not divided, and it is marked as "uncertain".

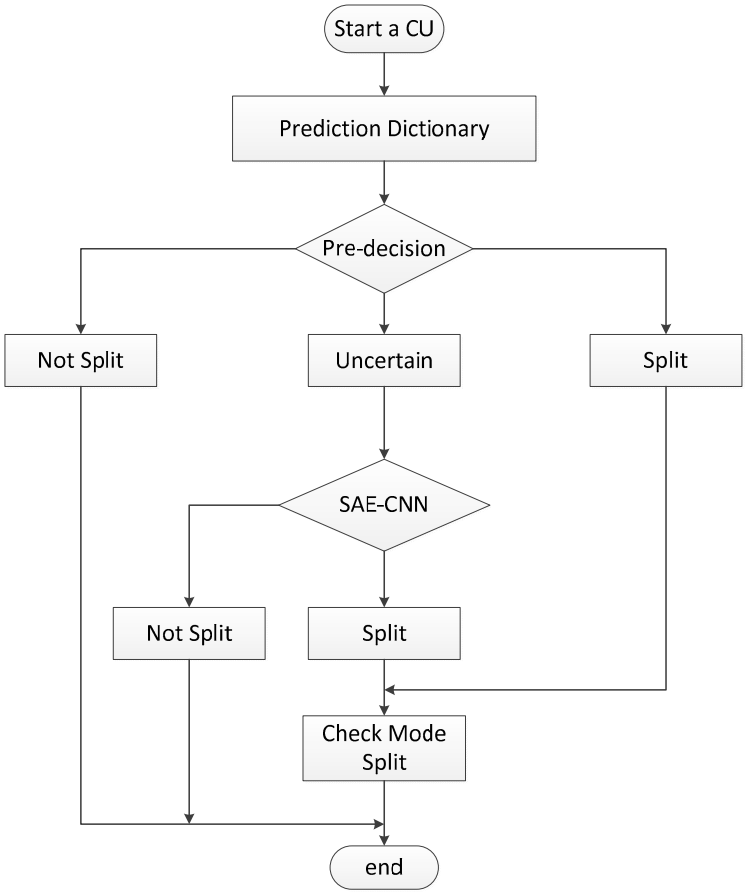



**Figure 3.** Flow chart of self-adaptive CU size partition decision.

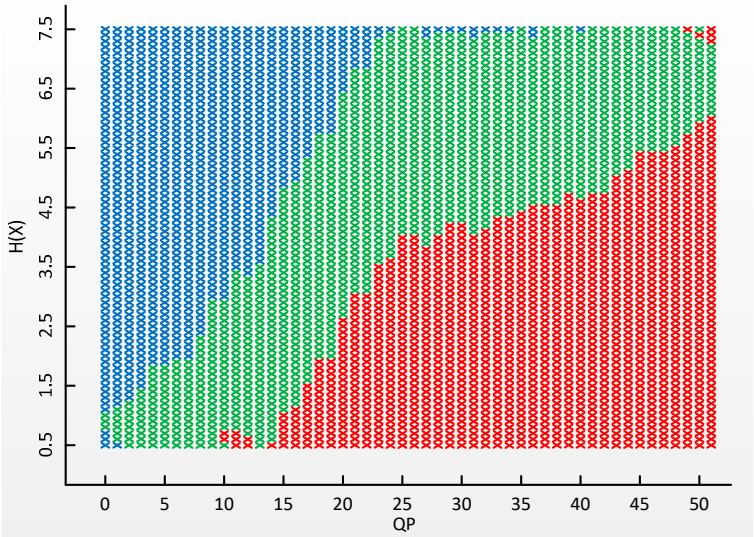



**Figure 4.** Prediction dictionary of CU partition.

Pre-decision division process: For a CU, the entropy value of the CU residual block is calculated, and the corresponding marking result can be found in the pre-decision dictionary according to the entropy value and the QP value of the current encoder. If it is marked as "split", the CU division mode selection is directly performed without SAE-CNN prediction, and the algorithm directly selects the division mode for the CU to complete the division process; if it is marked as "not split", the current CU division process can be

terminated early; if it is marked as "uncertain", then send the CU to SAE-CNN for division prediction; the above is the pre-decision process proposed in this article.

The pre-decision algorithm can avoid unnecessary SAE-CNN prediction and RD cost calculations for some CUs, determine their division results in advance and reduce coding time to a certain extent. In addition, the algorithm can filter out some CUs in the training samples that do not need to use SAE-CNN when SAE-CNN is training, thereby improving the prediction accuracy of the neural network.

### *3.2. SAE-CNN Architecture*

Since the QTMT division structure is adopted in the VVC, the size of the CU generated by the partition varies. If different CNN models of different sizes are designed to predict, not only will this result in a lot of unnecessary work, but it will also reduce the utilization of the model, resulting in a waste of resources. Therefore, it is necessary to design a CNN model that can predict CUs of different sizes. The pooling layer is well-known and an important part of the CNN architecture, and the pooling layer has no parameters that need to be learned, so the size of the pooling layer can be artificially defined. Due to this characteristic of the pooling layer, we propose a coding unit shape-adaptive CNN model. By adaptively adjusting the size of the pooling layer, CUs of different dimensions can be predicted using a CNN.

The SAE-CNN we propose includes an input layer, four convolutional layers, three pooling layers, a fully connected layer and an output layer. Specifically, the input is the residual block, and its size is recorded. Convolutional layers 1 and 2 (Conv1,2) contain 64 convolution kernels with a size of $5 \times 5$. In order to maintain the dimensions of the feature map, one-step convolution is performed on the input CU, and the obtained feature map is filled. Pooling layers 1 and 2 (Shape-adaptive Pool1,2) adopt the maximum pooling method, and their size is adaptively adjusted according to the size of the input CU. Convolutional layers 3 and 4 (Conv3,4) contain 64 $3 \times 3$ convolution kernels, which are similar to Conv1 and Conv2, with a stride of 1 and fill the feature maps. Pooling layer 3 (Pool3) adopts a maximum pooling with a size of $2 \times 2$. A fully connected layer (FC) contains 64 neurons, and the size of the input CU residual block and the corresponding quantization parameters are also joined as neurons in FC to improve the prediction accuracy; the output layer (SoftMax) uses the SoftMax function, which is composed of two neurons and outputs the probability of division and non-division, respectively, which is utilized to judge whether the input residual block is divided, thereby completing the entire decision-making process. In addition, the Rectified Linear Unit (Relu) is selected as the activation function of entire Conv. The biggest highlight of SAE-CNN is that regardless of the size of the input CU, the size of the feature map remains $4 \times 4$ before being sent to the FC, so that the SAE-CNN model can complete the division decision of CUs of different sizes. Figure 5 and Table 1 show the process and specific details of SAE-CNN mentioned in this article, respectively.

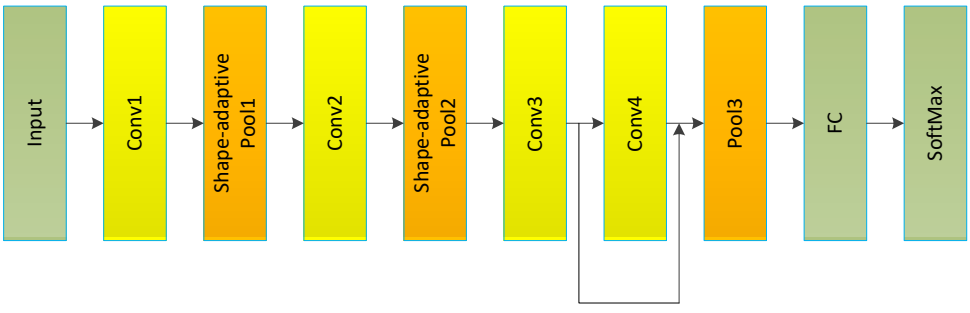



**Figure 5.** Diagram of proposed shape-adaptive CNN (SAE-CNN) structure.

**Table 1.** Details of the proposed shape-adaptive CNN (SAE-CNN) structure.

| Type | Structure |
|---|---|
| Input | Residual Block: Wide × Height × 1 |
| Conv1 | Filter: 5 × 5 × 64, Stride:1, Pad:2, ReLU |
| Shape-adaptive Pool1 | Max, Size X = (Wide ≥ 32) + 1, Size Y = (Height ≥ 32) + 1 |
| Conv2 | Filter: 5 × 5 × 64, Stride:1, Pad:2, ReLU |
| Shape-adaptive Pool2 | Max, Size X = (Wide ≥ 32) + 1, Size Y = (Height ≥ 32) + 1 |
| Conv3 | Filter: 3 × 3 × 64, Stride:1, Pad:1, ReLU |
| Conv4 | Filter: 3 × 3 × 64, Stride:1, Pad:1, ReLU |
| Pool3 | Max, Size X = 2, Size Y = 2 |
| FC | 64, include: QP, Wide, Height; ReLU |
| SoftMax | 2 |

As shown in Figure 6, taking the input CU size of 32 × 16 × 1 as an example, a 32 × 16 × 64 feature map is obtained after the Conv1. Then, according to W = 32 and H = 16, the shape-adaptive Pool1 becomes the largest pooling with a size of 2 × 1, and after this layer, a 16 × 16 × 64 feature map is obtained. After passing through the Conv2, the feature map size remains unchanged. In the same way, the shape-adaptive Pool2 becomes the maximum pooling with a size of 2 × 2, and the feature map size becomes 8 × 8 × 64 after this layer. Then, the feature maps obtained through the Conv3 and Conv4 are spliced together to obtain an 8 × 8 × 128 feature map. Finally, after the Pool3, the feature map size becomes 4 × 4 × 128. This also achieves the aforementioned highlights of the SAE-CNN we designed.

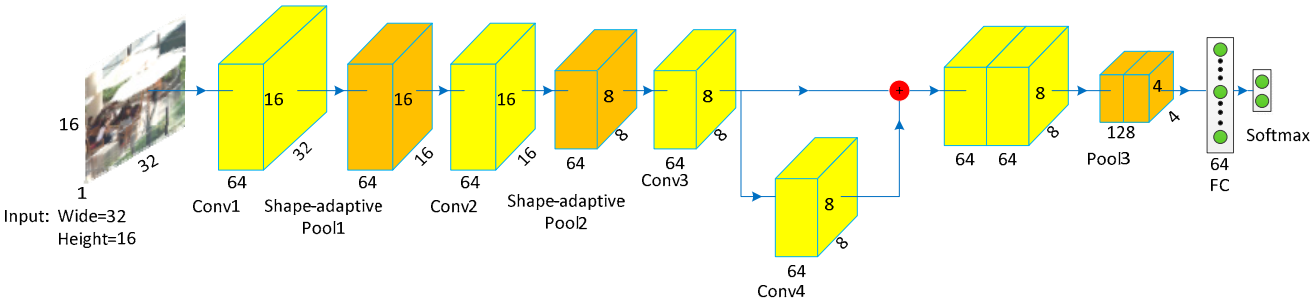



**Figure 6.** Example diagram with input 32 × 16 CU.

### *3.3. SAE-CNN Training*

The VVC standard test sequences selected to form the data set are shown in Table 2. These selected video sequences cover different textures, resolutions and scenes to ensure that the trained model has good universality. Each video randomly selects 25 frames and encodes them under different QPs. Among them, the first 20 frames form a train dataset to train the model; the last five frames form a test dataset to test the performance of SAE-CNN. After encoding, we extract CU residual blocks of different sizes and mark them according to the division of the CU. If the CU is divided, it is marked "1,0"; if it is not divided, it is marked "0,1". Since SAE-CNN can adapt to CUs of different sizes, if CUs of all sizes are formed by dataset according to the traditional training method, it is obvious that SAE-CNN cannot achieve a good training effect. Therefore, whether it is in the training dataset or test dataset, it must be divided into multiple datasets according to the size of the CU. Then, we use the pre-decision dictionary to filter out the obtained dataset, filter out some CUs in the sample that do not need to use SAE-CNN and obtain the final dataset. The cross-entropy function is used as the loss function during training, and the SAE-CNN

model is updated by Back Propagation (BP), and Stochastic Gradient Descent (SGD) is utilized for majorization. The formula of the loss function is as follows:

$$loss = \sum_{i=1}^{n} [y_i \log(\hat{y}) + (1 - y_i) \log(1 - \hat{y})] \quad (3)$$

$$\hat{y} = \frac{e^{x_i}}{\sum_{i=1}^{n} e^{x_j}} \quad (4)$$

where $y_i$ indicates the true value; $\hat{y}$ indicates the predicted value processed by the softmax function.

**Table 2.** Table of video sequence selection.

| Sequence | Class | Resolution |
|---|---|---|
| DaylightRoad2 | A | 3810 × 2160 |
| ArenaOfValor | B | 1920 × 1080 |
| BasketballDrillText | C | 832 × 480 |
| BasketballPass | D | 416 × 240 |
| Vidyo3 | E | 1280 × 720 |

We installed the SGD momentum and weight attenuation at 0.9 and 0.005 respectively, and installed the original learning rate at 0.001. We plan to decrease the learning rate by a coefficient of 0.25 after every 25,000 repetitions, and a total of 85,000 repetitions are performed to renew the SAE-CNN model. In addition, the training method is also different from the traditional method. The corresponding datasets composed of different CU sizes are sent to SAE-CNN for training in order from small to large. The 8 × 8 block will be trained foremost, and afterward the dataset with a size of 16 × 8 will be trained, and finally training datasets of different sizes will be trained in sequence. It should be noted that starting from the 16 × 8 dataset, the SAE-CNN parameters are obtained through training the 8 × 8 dataset, rather than randomly initialized parameters. By analogy, the SAE-CNN parameters are updated until the entire training process is completed.

## 4. Experimental Results

In this section, we describe the experiments conducted to assess the usefulness of our method in decreasing VVC encoding time. Here, we mainly introduce the experimental configuration and performance evaluation standards. At the same time, the experimental results are analyzed and compared with the current advanced technologies in terms of complexity reduction, bit rate loss and trade-off performance between coding complexity and coding quality.

### *4.1. Experimental Setup*

All experiments were carried out under the AI configuration in the VVC official test software VTM10.0 version. The CNN architecture was built and tested in python4.0 based on the pythorch learning library. Each encoding and SAE-CNN prediction was carried out individually on Intel Core i5-8500 processor running at 3.00 GHz on Window10 operating system. The CTC sequence specified by JVET is widely used because it contains a wide range of resolutions, texture complexities, bit depths and motion information. Therefore, we selected 15 sequences from the CTC sequence and divided them into five categories: A (3840 × 2160), B (1920 × 1080), C (832 × 480), D (416 × 240), and E (1280 × 720), forming a test set to test the proposed method. In the experiment, four QP (22,27,32,37) were used to encode each sequence separately, and the average value of the coding time reduction under four QP was calculated, and the experimental results were obtained. It should be noted that in order to ensure the fairness of the experiment and the effectiveness of the proposed algorithm, the sequence selected in the test set is completely different from the sequence selected in the CNN train dataset.

Here, we chose three standards to measure the performance of the proposed algorithm, including the Bjøntegaard Delta Bit Rate (BD-BR) to measure the coding quality; $\Delta T$ to measure the complexity reduction performance; and $\Delta T$/BD-BR to assess the trade-off performance of our method between coding complexity and coding quality. The calculation formula of $\Delta T$ is as follows:

$$\Delta T = \frac{T_{base} - T_{prop}}{T_{base}} \times 100\%, \quad (5)$$

where $T_{base}$ and $T_{prop}$ respectively represent the encoding time running in the original version of VTM10.0 and the encoding time running after the improved algorithm.

### 4.2. Results and Analysis

We contrasted the suggested algorithm with the method recommended by Tang et al. [25] and the method proposed by Li et al. [23]. It should be noted that [25] and [23] were tested under the VTM5.0 and VTM7.0 versions, respectively. In addition, the result selection in [23] comes from the running result of the "fast" mode in its proposed algorithm. The detailed experimental results are shown in Table 3.

**Table 3.** Coding performance of the proposed algorithm.

| Class | Sequence | Ref. [23], VTM7.0 | | | Ref. [25], VTM5.0 | | | Proposed Algorithm | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | BD-BR (%) | $\Delta T$ (%) | $\Delta T$/BD-BR | BD-BR (%) | $\Delta T$ (%) | $\Delta T$/BD-BR | BD-BR (%) | $\Delta T$ (%) | $\Delta T$/BD-BR |
| A | Campfire | 2.91 | 59.87 | 20.57 | 1.05 | 34.96 | 33.29 | 1.01 | 35.78 | 35.43 |
| | CatRobot1 | 3.28 | 55.99 | 17.07 | / | / | / | 0.96 | 36.98 | 38.52 |
| B | BQTerrace | 1.79 | 56.94 | 31.81 | 0.95 | 34.50 | 36.31 | 0.89 | 36.79 | 41.34 |
| | Cactus | 1.86 | 60.56 | 32.56 | / | / | / | 0.87 | 34.12 | 39.22 |
| | MarketPlace | 1.28 | 58.22 | 45.48 | / | / | / | 0.82 | 37.74 | 46.02 |
| | Kimono | / | / | / | 0.87 | 33.32 | 38.29 | 0.71 | 34.59 | 48.71 |
| C | BasketballDrill | 2.98 | 52.62 | 17.66 | 1.30 | 33.39 | 25.68 | 1.10 | 35.03 | 31.84 |
| | PartyScene | 1.16 | 58.94 | 50.81 | 0.55 | 31.10 | 56.54 | 0.67 | 34.55 | 51.57 |
| | RaceHorsesC | 1.61 | 57.89 | 35.96 | 0.37 | 23.63 | 63.86 | 0.75 | 33.89 | 45.19 |
| D | BlowingBubbles | 1.57 | 53.40 | 34.01 | 0.95 | 33.90 | 35.68 | 0.97 | 35.86 | 36.97 |
| | BQSquare | 1.33 | 55.16 | 41.47 | 0.68 | 30.73 | 45.19 | 0.71 | 32.35 | 45.56 |
| | RaceHorses | 1.88 | 53.34 | 28.37 | 0.71 | 31.79 | 44.77 | 0.76 | 33.14 | 43.61 |
| E | FourPeople | 2.20 | 59.74 | 27.15 | 1.38 | 38.01 | 27.54 | 0.99 | 38.40 | 38.79 |
| | KristenAndSara | 2.75 | 60.01 | 21.82 | 1.61 | 34.84 | 21.63 | 1.08 | 35.84 | 33.19 |
| | Video 1 | / | / | / | 1.63 | 38.73 | 23.76 | 1.32 | 38.93 | 29.49 |
| Average | | 2.05 | 57.13 | 27.87 | 1.00 | 33.24 | 33.24 | 0.91 | 35.60 | 39.12 |

The experimental results show that compared with the original VTM10.0 version, our suggested algorithm only increases the coding gain by 0.91% on average, but saves the coding time by 35.60%. In particular, after the sequence in Class E runs in the algorithm proposed in this article, it saves 37.72% coding time. Compared with [25], our experimental results have a smaller BD-BR increase and a larger coding time saving, which is more obvious in Figure 7. As shown in the figure, we randomly selected seven sequences to test with the method in [25] and the algorithm proposed in this paper, and the comparison results for coding time were obtained.

We used the algorithm proposed in this article and the original algorithm of VTM10.0 to encode the sequence *BasketballDrill* and randomly selected a frame to compare the CU division of the two, as shown in Figure 8. It can be seen that whether it is in the texture complex area (athlete area) or in the texture smooth area (floor area), the CU division shown by the algorithm proposed in this paper is mostly the same as the original algorithm. This also proves that the algorithm proposed in this paper can basically replace the original algorithm to complete the work of CU partitioning while reducing the coding complexity, and the partitioning result is close to the optimal partitioning result of the original algorithm.

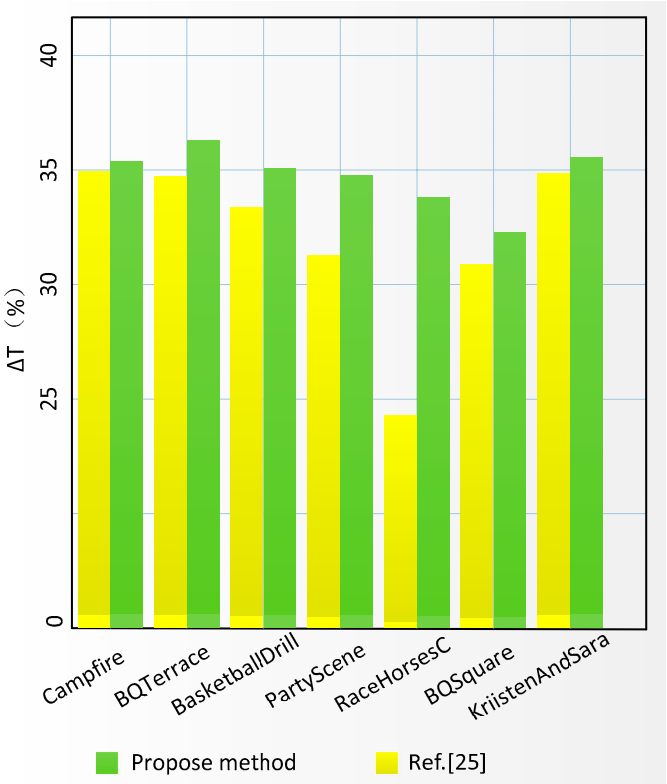



**Figure 7.** Histogram of experimental comparison.

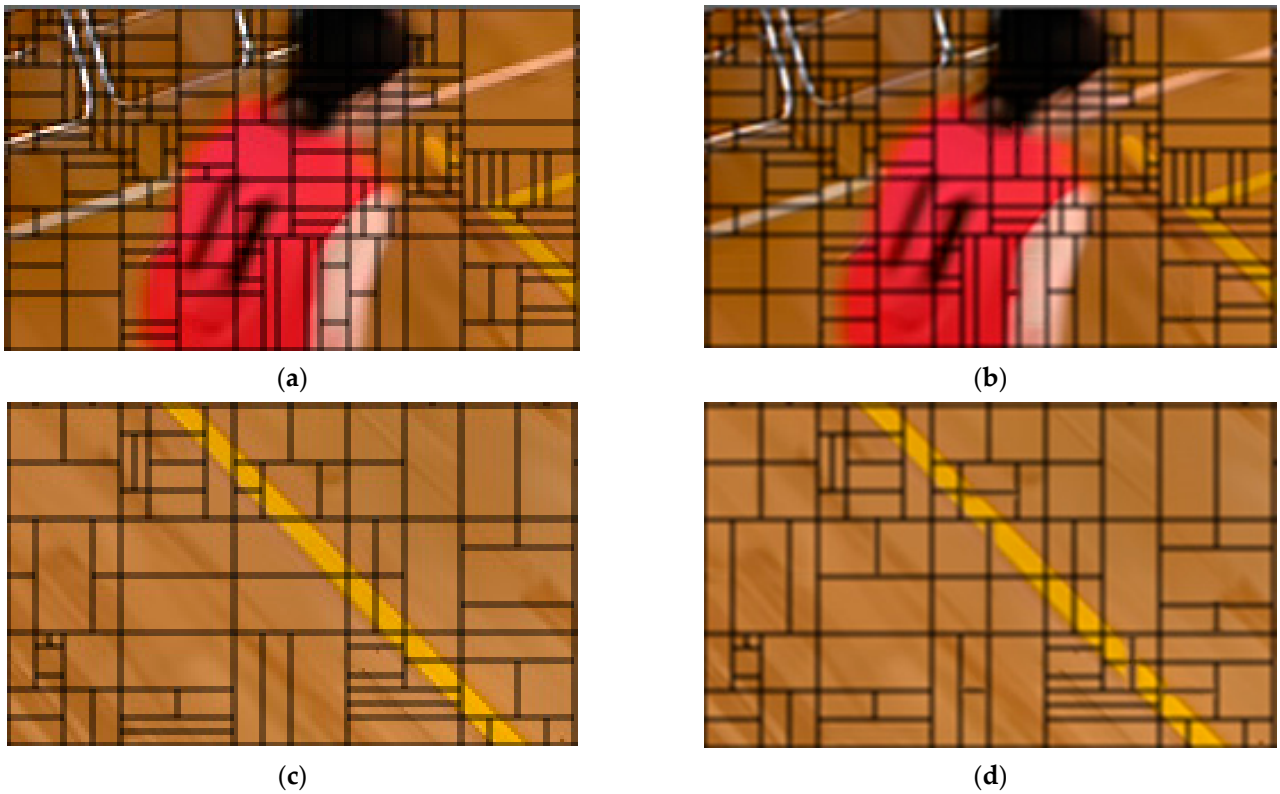



**Figure 8.** Chart of CU division results comparison: (**a**) CU partition map of complex texture area under VTM10.0 configuration; (**b**) CU partition map of complex texture area under the proposed algorithm; (**c**) CU partition map of smooth texture area under VTM10.0 configuration; and (**d**) CU partition map of complex texture area under the proposed algorithm.

In addition, compared with [23], although the algorithm recommended in this article is not as effective as [23] in terms of saving coding time, the algorithm of [23] caused a substantial increase in BD-BR in order to increase the saving of coding time. So, the algorithm of [23] does not balance the relationship between coding complexity and coding quality. It is also obvious from Table 3 that the average value of ΔT/BD-BR obtained by our proposed algorithm is 39.12, and the average value of ΔT/BD-BR in [23] is only 27.87. Therefore, our proposed algorithm can achieve a better balance between coding complexity and coding quality. In summary, the algorithm suggested in this article has improved compared with the original method and other algorithms in terms of reducing the coding complexity or balancing the reduction of complexity and the increase in bit rate.

## 5. Conclusions

This article proposes a VVC intra prediction complexity reduction algorithm based on statistical theory and SAE-CNN. The proposed algorithm was applied to the residual block. We used pre-decision and SAE-CNN to predict whether the coding unit is divided and if part of RDO can be reduced, so as to attain the aim of decreasing coding time. In addition, unlike the traditional CNN architecture, we introduced the concept of a shape-adaptive pooling layer in the CNN architecture. Additionally, in the data set establishment and training plan, we propose the establishment of a corresponding size data set according to the different CU sizes and training the CNN in the order of the size of the CU from small to large, so as to achieve the purpose of using a CNN to divide CUs of different sizes and improve the network utilization. Compared with the original VTM10.0, our proposed algorithm reduces the coding complexity by 35.60%, while BD-BR only increases by 0.91%; Compared with other advanced technologies, our proposed algorithm can also achieve a better balance between the reduction in coding complexity and the increase in BD-BR. This also proves that the algorithm proposed in this paper has a good performance.

**Author Contributions:** Conceptualization, J.Z. and P.D.; methodology, J.Z.; software, P.D.; validation, J.Z., Q.Z. and P.D.; formal analysis, P.D.; investigation, P.D.; resources, Q.Z.; data curation, P.D.; writing—original draft preparation, P.D.; writing—review and editing, J.Z.; visualization, J.Z.; supervision, Q.Z.; project administration, Q.Z.; funding acquisition, Q.Z. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the National Natural Science Foundation of China, grant number 61771432, and 61302118, the Basic Research Projects of Education Department of Henan, grant number 21zx003, and 20A880004 and the Postgraduate education reform and quality improvement project of Henan Province, grant number YJS2021KC12.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

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

## References

1. Ouahabi, A. *Signal and Image Multiresolution Analysis*; ISTE-Wiley: Hoboken, NJ, USA, 2013.
2. Ouahabi, A. A review of wavelet denoising in medical imaging. In Proceedings of the 2013 8th International Workshop on Systems, Signal Processing and their Applications (WoSSPA), Algiers, Algeria, 12–15 May 2013; pp. 19–26.
3. Ohm, J.; Sullivan, G.J.; Schwarz, H.; Tan, T.K.; Wiegand, T. Comparison of the Coding Efficiency of Video Coding Standards—Including High Efficiency Video Coding (HEVC). *IEEE Trans. Circuits Syst. Video Technol.* **2012**, *22*, 1669–1684. [CrossRef]
4. Ferroukhi, M.; Ouahabi, A.; Attari, M.; Habchi, Y.; Taleb-Ahmed, A. Medical Video Coding Based on 2nd-Generation Wavelets: Performance Evaluation. *Electronics* **2019**, *8*, 88. [CrossRef]
5. Bross, B. Versatile Video Coding (Draft 1). JVET-J1001. 2018. Available online: http://phenix.it-sudparis.eu/jvet/doc_end_user/current_document.php?id=3489 (accessed on 10 October 2021).
6. Bross, B.; Chen, J.; Ohm, J.-R.; Sullivan, G.J.; Wang, Y.-K. Developments in International Video Coding Standardization After AVC, With an Overview of Versatile Video Coding (VVC). *Proc. IEEE* **2021**, *109*, 1463–1693. [CrossRef]

7. Bross, B.; Chen, J.; Liu, S. Versatile Video Coding (Draft 7). JVET-P2001. 2019. Available online: http://phenix.it-sudparis.eu/jvet/doc_end_user/current_document.php?id=8857 (accessed on 10 October 2021).
8. Song, X.D.; Fan, X.C.; Xiang, C.C.; Ye, Q.W.; Liu, L.Y.; Wang, Z.W.; He, X.J.; Yang, N.; Fang, G.F. A Novel Convolutional Neural Network Based Indoor Localization Framework with WiFi Fingerprinting. *IEEE Access* **2019**, *7*, 110698–110709. [CrossRef]
9. Li, S.; Dai, W.; Zheng, Z.; Li, C.; Zou, J.; Xiong, H. Reversible Autoencoder: A CNN-Based Nonlinear Lifting Scheme for Image Reconstruction. *IEEE Trans. Signal Process.* **2021**, *69*, 3117–3131. [CrossRef]
10. Ouahabi, A.; Taleb-Ahmed, A. Deep learning for real-time semantic segmentation: Application in ultrasound imaging. *Pattern Recognit. Lett.* **2021**, *144*, 27–34. [CrossRef]
11. Heindel, A.; Haubner, T.; Kaup, A. Fast CU split decisions for HEVC inter coding using support vector machines. In Proceedings of the 2016 Picture Coding Symposium (PCS), Nurnberg, Germany, 4–7 December 2016; pp. 1–5.
12. Zhang, Y.; Kwong, S.; Wang, X.; Yuan, H.; Pan, Z.; Xu, L. Machine Learning-Based Coding Unit Depth Decisions for Flexible Complexity Allocation in High Efficiency Video Coding. *IEEE Trans. Image Process.* **2015**, *24*, 2225–2238. [CrossRef] [PubMed]
13. Xu, C.; Liu, P.; Wu, Y.; Jia, K.; Dong, W. A Fast CTU Depth Selection Algorithm for H.265/HEVC Based on Machine Learning. In Proceedings of the 2018 IEEE 3rd International Conference on Signal and Image Processing (ICSIP), Shenzhen, China, 13–15 July 2018; pp. 154–161.
14. Jamali, M.; Coulombe, S.; Sadreazami, H. CU Size Decision for Low Complexity HEVC Intra Coding based on Deep Reinforcement Learning. In Proceedings of the 2020 IEEE 63rd International Midwest Symposium on Circuits and Systems (MWSCAS), Springfield, MA, USA, 9–12 August 2020; pp. 586–591.
15. Kim, K.; Ro, W.W. Fast CU Depth Decision for HEVC Using Neural Networks. *IEEE Trans. Circuits Syst. Video Technol.* **2019**, *29*, 1462–1473. [CrossRef]
16. Zhang, G.; Xiong, L.; Lian, X.; Zhou, W. A CNN-based Coding Unit Partition in HEVC for Video Processing. In Proceedings of the 2019 IEEE International Conference on Real-Time Computing and Robotics (RCAR), Irkutsk, Russia, 4–9 August 2019; pp. 273–276.
17. Guo, X.; Wang, Q.; Jiang, J. A Lightweight CNN for Low-Complexity HEVC Intra Encoder. In Proceedings of the 2020 IEEE 15th International Conference on Solid-State & Integrated Circuit Technology (ICSICT), Kunming, China, 3–6 November 2020; pp. 1–3.
18. Fu, T.; Zhang, H.; Mu, F.; Chen, H. Fast CU Partitioning Algorithm for H.266/VVC Intra-Frame Coding. In Proceedings of the 2019 IEEE International Conference on Multimedia and Expo (ICME), Shanghai, China, 8–12 July 2019; pp. 55–60.
19. Cui, J.; Zhang, T.; Gu, C.; Zhang, X.; Ma, S. Gradient-Based Early Termination of CU Partition in VVC Intra Coding. In Proceedings of the 2020 Data Compression Conference (DCC), Snowbird, UT, USA, 24–27 March 2020; pp. 103–112.
20. Wu, G.; Huang, Y.; Zhu, C.; Song, L.; Zhang, W. SVM Based Fast CU Partitioning Algorithm for VVC Intra Coding. In Proceedings of the 2021 IEEE International Symposium on Circuits and Systems (ISCAS), Daegu, Korea, 22–28 May 2021; pp. 1–5.
21. Tissier, A.; Hamidouche, W.; Vanne, J.; Galpin, F.; Menard, D. CNN Oriented Complexity Reduction of VVC Intra Encoder. In Proceedings of the 2020 IEEE International Conference on Image Processing (ICIP), Abu Dhabi, United Arab Emirates, 25–28 October 2020; pp. 3139–3143.
22. Galpin, F.; Racapé, F.; Jaiswal, S.; Bordes, P.; Le Léannec, F.; François, E. CNN-Based Driving of Block Partitioning for Intra Slices Encoding. In Proceedings of the 2019 Data Compression Conference (DCC), Snowbird, UT, USA, 26–29 March 2019; pp. 162–171.
23. Li, T.; Xu, M.; Tang, R.; Chen, Y.; Xing, Q. DeepQTMT: A Deep Learning Approach for Fast QTMT-Based CU Partition of Intra-Mode VVC. *IEEE Trans. Image Process.* **2021**, *30*, 5377–5390. [CrossRef] [PubMed]
24. Wang, Z.; Wang, S.; Zhang, X.; Wang, S.; Ma, S. Fast QTBT Partitioning Decision for Interframe Coding with Convolution Neural Network. In Proceedings of the 2018 25th IEEE International Conference on Image Processing (ICIP), Athens, Greece, 7–10 October 2018; pp. 2550–2554.
25. Tang, G.; Jing, M.; Zeng, X.; Fan, Y. Adaptive CU Split Decision with Pooling-variable CNN for VVC Intra Encoding. In Proceedings of the 2019 IEEE Visual Communications and Image Processing (VCIP), Sydney, Australia, 1–4 December 2019; pp. 1–4.
26. Huang, Y.-H.; Chen, J.-J.; Tsai, Y.-H. Speed Up H.266/QTMT Intra-Coding Based on Predictions of ResNet and Random Forest Classifier. In Proceedings of the 2021 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 10–12 January 2021; pp. 1–6.
27. Fu, P.-C.; Yen, C.-C.; Yang, N.-C.; Wang, J.-S. Two-phase Scheme for Trimming QTMT CU Partition using Multi-branch Convolutional Neural Networks. In Proceedings of the 2021 IEEE 3rd International Conference on Artificial Intelligence Circuits and Systems (AICAS), Washington, DC, USA, 6–9 June 2021; pp. 1–6.