

Cost-Efficient HEVC-Based Quadtree Splitting (HEQUS) for VVC Video Transcoding

D. García-Lucas^{a,*}, G. Cebrián-Márquez^b, A. J. Díaz-Honrubia^c, T. Mallikarachchi^d and P. Cuenca^a

^aHigh Performance Networks and Architectures, University of Castilla-La Mancha, Albacete, Spain

^bDepartment of Computer Science, University of Oviedo, Oviedo, Spain

^cETS de Ingenieros Informáticos, Universidad Politécnica de Madrid, Madrid, Spain

^dSchool of Technologies, Cardiff Metropolitan University, Cardiff, United Kingdom

ARTICLE INFO

Keywords:

HEVC

H.265

VVC

Transcoding

MTT

ABSTRACT

With the advent of the new video coding standard Versatile Video Coding (VVC), there is now a need for converting current multimedia content from the High Efficiency Video Coding (HEVC) standard to the new format. VVC is focused on ultra-high definition (UHD) content, and is expected to be ready by 2020. Therefore, a traditional transcoding pipeline composed of a sequential cascaded HEVC decoder and a VVC encoder is not effective due to the exorbitant computational complexity of VVC. In this regard, this paper proposes a fast HEVC-VVC transcoder composed of a probabilistic classifier based on Naïve-Bayes in the first partitioning level (128×128 pixels), and a model that determines the partitioning of subsequent VVC coding depth levels on the basis of the HEVC partitioning. The proposed Naïve-Bayes classifier uses the features extracted from the 128×128 size blocks of the residual and reconstructed frames in HEVC, and their correlation with the block partitioning structure. The evaluation of the algorithm shows that it achieves a 44.07% time saving with a BD-rate penalty of 2.11% in the random access scenario.

1. Introduction

The High Efficiency Video Coding (HEVC) standard is the latest video coding project launched on the market by the Joint Collaborative Team on Video Coding (JCT-VC) in 2013 [16]. HEVC is gradually replacing its predecessor, the H.264/Advanced Video Coding (AVC) standard, which has been the most widely used codec in recent years in many applications, such as streaming or broadcasting [17]. On average, HEVC doubles the compression performance of H.264/AVC, especially for high definition (HD) and ultra-high definition (UHD) content, but at the cost of a great increase in processing times [24].

The most recent user demands introduce new challenges that require even more efficient compression techniques, since 75% of total Internet traffic is video traffic, and is predicted to reach 82% in 2022 [7]. For this reason, the international organizations ITU-T, through the Video Coding Expert Group (VCEG), and ISO/IEC, through the Moving Picture Expert Group (MPEG), have jointly created a new collaboration framework under the name of Joint Video Experts Team (JVET). Since the creation of the JVET

Table 1

VTM 2.0.1 performance over HM 16.19 [26].


Scenario	PSNR (%)			Time increase (%)	
	Y	U	V	Encoder	Decoder
All Intra	−18.0	−23.7	−24.5	1812	166
Random Access	−23.1	−34.6	−33.0	373	126
Low Delay B	−18.3	−27.6	−28.1	317	129
Low Delay P	−21.9	−30.7	−31.4	288	132

group in October 2015, the development of the new coding standard began under the name of Versatile Video Coding (VVC), with a compression capability that significantly surpasses the one achieved by HEVC, especially for streaming UHD, panorama video, sports events, concerts, 360° omnidirectional immersive multimedia and high-dynamic-range (HDR) video content.

Table 1 shows a comparison of both standards through their reference encoding and decoding software, namely HEVC Test Model (HM) [19], and VVC Test Model (VTM) [20]. It should be noted that VVC improves the compression performance of HEVC, yet at the same time, introduces high computational cost in the encoding process. Therefore, considering the superior compression performance of VVC and, at the same time, the large amount of existing content currently encoded under HEVC, a transcoder that converts bitstreams from HEVC to VVC is of great value for many applications, providing interoperability between the HEVC standard and the format of the future video compression standard. However, due to the high computational cost of VVC, a simple cascade transcoding system composed

This document is the results of the research project supported by the Spanish Ministry of Science, Innovation and Universities, and the European Commission (FEDER funds) under project RTI2018-098156-B-C52, by the Regional Government of Castilla-La Mancha under project SBPLY/17/180501/000353, and by the Spanish Ministry of Education, Culture and Sports under Grant FPU16/05692.

*Corresponding author

 david.garcialucas@uclm.es (D. García-Lucas);

cebriangabriel@uniovi.es (G. Cebrián-Márquez);

antoniojesus.diaz@upm.es (A. J. Díaz-Honrubia);

tmallikarachchi@cardiffmet.ac.uk (T. Mallikarachchi);

pedro.cuenca@uclm.es (P. Cuenca)

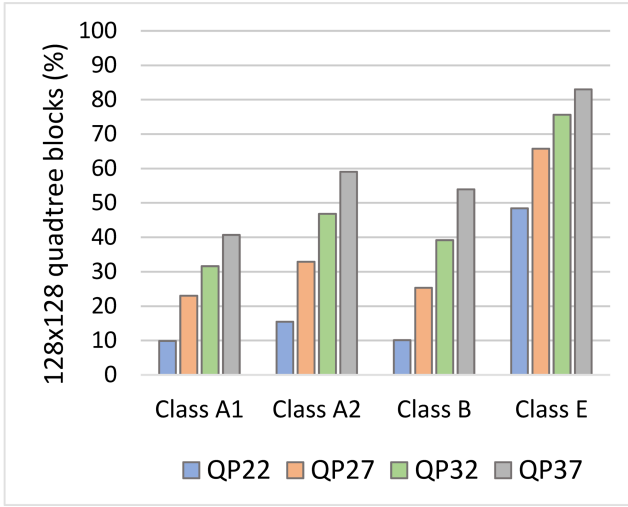


Figure 1: Average percentage of unsplit 128×128 blocks per class and QP.

of an HEVC decoder and a VVC encoder is not sufficiently effective. Therefore, strategies for accelerating the HEVC to VVC transcoding process are needed.

In this context, this work presents a fast transcoding approach between HEVC and VVC, called HEVC-based QUadtree Splitting (HEQUS) algorithm, which aims to exploit the information gathered in the HEVC decoder in order to assist decisions on the VVC encoder quadtree (QT) partitioning, taking advantage of the similarities in the splitting scheme between the two standards. The maximum block size is 128×128 pixels in VVC, while in HEVC the maximum size is 64×64 pixels. Therefore, there is no direct relationship between a block of 128×128 pixels in VVC and any block in HEVC. In this regard, Fig. 1 shows the percentage of 128×128 blocks that remain unsplit in VVC with respect to the quantization parameter (QP) for high-resolution sequences [21]. This highlights the relevance of the new block size, especially in low-bitrate scenarios.

On the basis of the above, the present work introduces a probabilistic model based on Naïve-Bayes to analyze statistical information obtained from the HEVC bitstream with the aim of accelerating the splitting decision of 128×128 pixel blocks. Since the most demanded applications are on-demand video and live streaming, which use the random access (RA) scenario, the Naïve-Bayes model is built from information of sequences encoded under RA configuration. For subsequent levels, i.e., block sizes ranging from 64×64 to 8×8 pixels, the QT partitioning structures of VVC and HEVC share many similarities. Therefore, the splitting decisions in the QT structure of VVC can be taken on the basis of the coding unit (CU) partitioning of HEVC, avoiding the brute force scheme of rate-distortion optimization (RDO).

The experimental results show that the proposed algorithm achieves time savings of up to 59.13% on average using the full set of the JVET common test sequences compared with the anchor cascade transcoder [21]. With respect to the coding efficiency of the proposed scheme, the results

show a penalty lower than 3.15% in terms of the Bjøntegaard delta rate (BD-rate), which measures the increment in bitrate while maintaining the same objective video quality [2].

The remainder of this paper is organized as follows. Section 2 includes relevant related work. Section 3 highlights key features of the VVC coding design compared to HEVC. The proposal is described in Sect. 4, and an analysis of the results is presented in Sect. 5. Finally, Sect. 6 concludes the paper by proposing possible lines of future work.

2. Related Work

The topic of video content transcoding between standards, also called heterogeneous transcoding, has been widely studied, given that the simplest transcoding process, i.e., cascade transcoding, is not effective because of the high computational cost involved [28]. A number of proposals in the literature include fast transcoders that accelerate the process of transcoding through different techniques. In this section, some transcoding studies based on similarity of the coding structure between different standards are analyzed.

In [8], a complete transcoding algorithm between standards H.264/AVC and HEVC was presented by Diaz-Honrubia et al. in 2016. A total of 8 probabilistic models were built based on a Naïve-Bayes classifier for each level of partitioning (64×64 and 32×32 pixels) and temporal layer. In addition, each model was constructed with the information of 26 variables extracted from the decoder of H.264/AVC. These variables were calculated for 1,000 instances for each of the 4 sequences trained and QPs. The full implementation of the algorithm achieved a quantitative speed-up of around 2.31× on average, with a time reduction of 56.7% and a BD-rate penalty of around 3.4%, compared with the anchor transcoder.

In 2017, X. Li et al. proposed four models for different depth and QP values using Naïve-Bayes classifiers implemented in a machine-learning-based VP9 to HEVC transcoder [22]. VP9 is an open source royalty-free codec developed by Google to join the competition for video coding, and was mainly designed for web video applications scenarios, as it is supported by multiple browsers for practical consumer use [11]. VP9 divides large coding units of 64×64 all the way down into 4×4 blocks. This, along with subblock counting and depth map, is the input information for the trained models. With an averaged BD-rate penalty of 2.8%, this proposal achieves a 44% time reduction compared with the full VP9-HEVC transcoder.

In 2018, J.-F. Franche and S. Coulombe proposed a fast H.264/HEVC transcoder composed of a motion propagation algorithm and a fast mode decision framework [10]. The motion propagation algorithm creates a motion vector candidate list at coding tree unit (CTU) level, and then selects the best candidate at prediction unit (PU) level. By pre-computing the prediction error of each candidate at CTU level and by reusing the information for various partition sizes, this method avoids computational redundancies. The fast mode decision framework is based on a post-order traversal of the

CTU, which includes several mode reduction techniques. Moreover, a novel method exploits the data provided by the motion propagation algorithm to determine whether a CU has to be split. Compared with a cascaded pixel-domain transcoding approach, this solution is on average 8.5× faster using one reference frame with a 2.63% BD-rate penalty. For a configuration with four reference frames, the average speed-up is 11.77× and the penalty is 3.82% BD-rate.

In [23], a moving object tracker is used to track objects in input compressed domain and remove the need to do most of the complex inter prediction in an H.264/H.265 to AV1 transcoder. In this proposal, presented in 2018, the encoder motion estimation block is supplied with a refined motion vector field from an object tracker that can skip a massive portion of the inter prediction module, which results in a significant reduction in coding time. The results estimate that the run time achieved by reusing motion data is between 3 and 4 times lower than that of the anchor transcoder, but with a greater visual quality decrease.

Finally, a transcoder based on CU depth inheritance between HEVC and AV1 was presented in 2019 [3]. AV1 is the codec developed by AOMedia group, in which different companies are involved in the creation of a royalty-free video coding format [1]. In this proposal, a set of experiments were performed aiming to identify possible correlations between CUs and block size decisions performed by the two codecs. After this analysis, the proposal consists in inheriting the decoded CU size information from the HEVC bitstream to infer decisions taken during the AV1 re-encoding process. When the proposal is compared to the unmodified AV1 transcoder flow, an average time saving of 35.41% is achieved, with a compression efficiency loss of 4.54% in terms of BD-rate.

With the analysis of these works, some aspects can be improved. For example, in the case of [8] and [22], multiple prediction models are defined. These proposals can be improved by using the lambda (λ) encoding parameter, which depends on the QP and temporal layer of the frame, simplifying these models into a single model with good results. In addition, some of these works introduce high BD-rate penalties in the transcoding process, which should be avoided in order to have a good compression performance.

The features, models and algorithms analyzed in the literature are specific to the standards involved in each proposal, and are thus not applicable to an HEVC-VVC transcoder. For this reason, this article presents a new algorithm for fast transcoding between HEVC and VVC. To the best of the authors' knowledge, there are no other HEVC-to-VVC transcoding schemes in the state of the art at the moment of writing. Therefore, it opens a new line of research in this field.

3. Technical Background

The development of VVC began with the Joint Exploration Test Model (JEM) [5], which was in turn built on top of HM, thus inheriting many features and coding tools from previous standards. VVC has introduced many new encoding tools and has improved existing ones [4]. As a re-

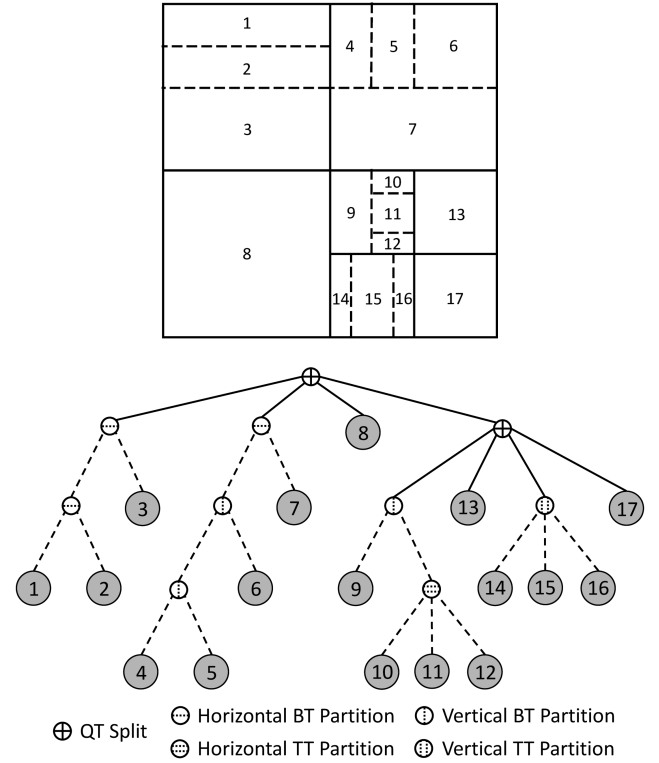


Figure 2: Example of an MTT structure.

sult, while VVC maintains the hybrid block-based scheme of HEVC, it achieves significant improvements in terms of coding efficiency. This section highlights the details of the tools introduced in VVC, comparing them to HEVC.

Regarding the block partitioning structure, HEVC introduced the so-called CTUs, which are square regions of up to 64×64 pixels that can be recursively split into four square-shaped CUs down to 8×8 pixels using a QT structure. Each CU is encoded using either inter-picture (temporal) or intra-picture (spatial) prediction. In this regard, each block can contain one or more PUs, and is encoded using a QT of transform units (TUs). VVC, in turn, increases the maximum CTU size to 128×128 pixels, and enables multiple possible partitioning schemes in what is known as a multi-type tree (MTT) structure, leaving the concepts of PUs and TUs aside. MTT is based on two splitting stages: firstly, a QT is used to split a CU into four sub-CUs of equal size, and secondly, the leaf nodes of the QT are split horizontally or vertically by the use of binary trees (BTs) and ternary trees (TTs), respectively. In the case of leaf nodes in QT of size 128×128 pixels, only the BT partitioning is evaluated, since the TT is not allowed at this level. An example of MTT partitioning is shown in Fig. 2, which shows that this approach makes the encoder much more adaptive to the local characteristics of the input video sequence.

With regard to intra prediction, VVC introduces 65 prediction modes, which are depicted in Fig. 3, as opposed to the 33 modes defined by HEVC. This extension of directional modes allows us to capture finer edge directions present in natural videos in both luma and chroma compo-

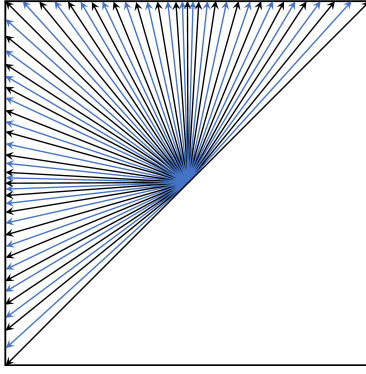


Figure 3: Intra prediction modes in VVC.

nents, whereas the Planar and DC modes remain the same. Another technique incorporated in the intra module is called the cross-component linear model (CCLM), which reduces redundancies between luma and chroma components, considering that VVC enables separate chroma encoding from the luma component.

As far as inter prediction is concerned, the VVC affine motion compensated prediction has been extended to two motion compensation models, based on a 4-parameter model, which uses information from two motion vectors (MVs), and a 6-parameter model that uses up to three MVs. The MV of the center sample of each sub-block is calculated according to these models, using 1/16 fraction accuracy. In addition, the subblock-based temporal MV prediction (SbTMVP) technique is similar to the temporal MV prediction (TMVP) tool used in HEVC, but has two main differences. On the one hand, the motion prediction is performed at sub-CU level. On the other hand, SbTMVP applies a motion shift before the temporal motion information is referenced in the collocated picture. The last tool implemented in this module is the adaptive MV resolution (AMVR), where motion vector differences can be coded in units of quarter-luma-sample, integer-luma-sample or four-luma-sample. To determine the motion vector resolution for the current CU, the encoder performs rate-distortion checks.

In VVC, a high-precision MV storage is included, with up to 1/16 fraction accuracy for merge, affine and MV storage. In this regard, in AMVR, for which the highest accuracy is 1/4, the precision is also shifted to 1/16.

Finally, regarding the modifications to the transform, which allows the encoder to compact the residual information, VVC defines a multiple transform selection (MTS) procedure, which is used for residual coding both inter and intra coded blocks. With this tool, the encoder can choose among a set of different transform functions, namely DCT-II, DCT-VIII and DST-VII.

4. Proposed Transcoding Approach

This section describes the HEVC-VVC transcoding approach using the HEQUS algorithm. At the first partitioning level, a probabilistic model based on Naïve-Bayes classifiers

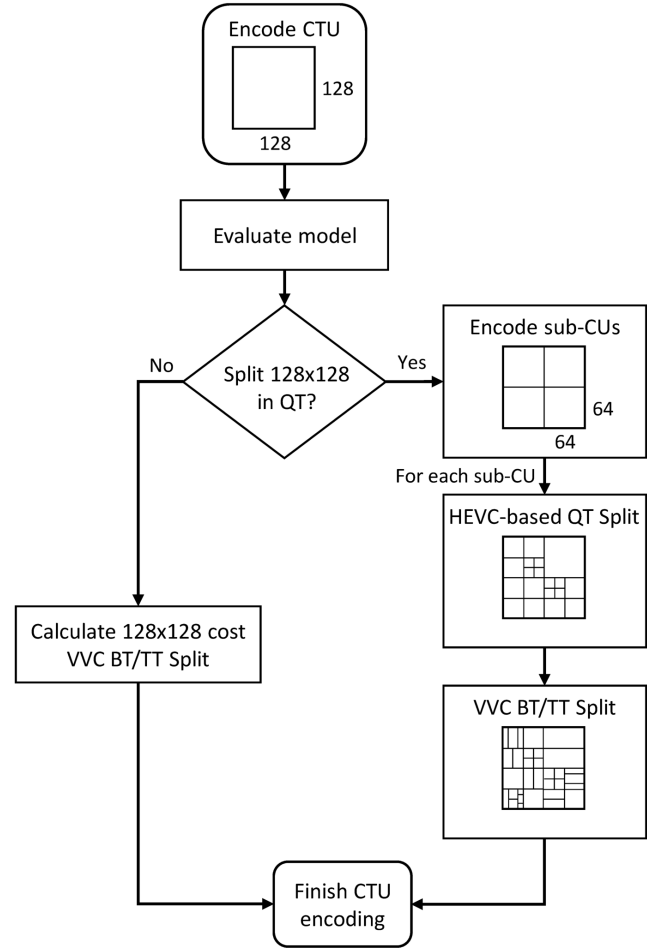


Figure 4: Encoding process of the HEQUS algorithm in the proposed HEVC-VVC transcoder.

is used to assist the splitting decision on the basis of previously coded information and statistical data. At the remaining levels, the QT partitioning used in VVC is based on the block structure used in HEVC.

4.1. Description of the HEQUS Algorithm

This proposal aims to speed up the decision making of the QT partitioning in the context of an HEVC-VVC transcoding scenario to replace the brute-force scheme used in typical encoder implementations. However, the first partitioning level in VVC involves 128×128 pixel blocks, while the maximum block size in HEVC is 64×64 pixels. For this reason, there is no direct relationship between the first level of the VVC partitioning and partitioning carried out by HEVC. To address this, the proposed approach implements a mathematical model to predict the partitioning of the first level in the transcoding process. This approach, which is depicted in Fig. 4, is detailed step by step in the following paragraphs.

The source HEVC video stream contains valuable information that can be used as features and statistical information following a knowledge discovery from data (KDD) approach [9]. In this process, the decoded frames and their cor-

responding residual information are divided into 128×128 pixel blocks to extract useful information for the transcoding. This information is preprocessed and later processed using machine learning techniques to create a decision model that accelerates the transcoding by deciding whether the 128×128 blocks must be split or not. If the model decides to split the block, it is divided into 4 CUs of 64×64 pixels each, thus reducing the total computation time, since the evaluation of QT and BT is skipped for this level. On the contrary, if the model decides not to split the 128×128 block, the evaluation of lower levels in the partitioning tree is completely omitted, resulting in significant time savings.

To prevent the evaluation of the entire MTT structure for the second partitioning level onward when a 128×128 block is split, our transcoding approach makes use of the original HEVC partitioning structure to accelerate the evaluation of 64×64 , 32×32 , 16×16 and 8×8 CUs. In this regard, our proposed HEQUS algorithm is based on the QT structure of HEVC so that the VVC encoder only needs to evaluate the BT and the TT of the resulting tree structure. However, the VTM encoder implementation includes early skip techniques, and, given that the decisions of the HEQUS algorithm are implemented on top of these techniques of the VVC encoder, a second implementation called Fast HEQUS is presented, where this method overrides the fast checks in QT blocks, so that the QT partitioning adopted by the transcoder in 64×64 blocks and lower is completely identical to that of HEVC.

The following subsections outline the steps carried out in the development of the proposal. The first step consists on data understanding by describing the features used to construct the model of the first partitioning level, the generation of the training model, and the model building process itself. Additional subsections, in turn, indicate how the HEVC partitioning structure is used in VVC to accelerate the QT partitioning of the remaining levels.

4.2. Data Understanding

Features are individual independent variables that act as input information for decision models to make predictions. In the case of our proposal, this information is taken from the HEVC stream or from the transcoding process itself, and is used to predict the partitioning decision of the 128×128 blocks in VVC. In this regard, we selected a large set of features, numbered from V_1 to V_{16} , that potentially describe the distinctive characteristics of the input sequences, and thus help predict the partitioning of the first level [14]. Many of them rely on the existing correlation between the texture of a region or its residual, and its most efficient partitioning. Among the proposed features, some are known to provide useful information in a transcoding process [8, 25, 12]. The initial set of features contains the following variables:

- Average of the block (\bar{x}): calculated for the samples in the 128×128 residual block (V_1), which can describe the complexity of the prediction obtained for the current block.

- Variance of the block (σ^2): variance of the samples in the 128×128 block, both in the residual frame (V_2) and in the reconstructed image (V_9).
- Variance of the means in sub-blocks: since the QT divides a block into four sub-blocks of equal size, it is interesting to know information about these sub-blocks. Thus, the 128×128 residual block is divided into four blocks of size 64×64 . The mean of the residual values of each 64×64 is calculated, and then the variance of these means (V_3).
- Variance of the variances in sub-blocks: similar to the previous statistic, the 128×128 residual block is divided into four blocks of size 64×64 . In this case, variance of the residual values of each 64×64 block is calculated, and then the variance of these variances (V_4).
- Fisher coefficient of skewness (γ): measure of skewness, or more precisely, the lack of symmetry, of a set of values based on their distribution around the average. In the case of a normal distribution, the skewness must be close to zero, or on the contrary the distribution is asymmetric to the left or right. This statistic has been calculated for the 128×128 block in both the residual frame (V_5) and the reconstructed image (V_7). The following expression represents γ , where P is the value of each sample and N is the total of samples contained in a block of size 128×128 pixels.

$$\gamma = \frac{\sum_{i=1}^N (P_i - \bar{x})^3}{N \cdot \sigma^3}$$

- Mean absolute deviation (MAD): this feature shows the amount of deviation that occurs around the mean in a set of values by calculating the average distance between each value and the central value. It has also been calculated for the 128×128 block in both the residual frame (V_6) and the reconstructed image (V_8).

$$MAD = \frac{\sum_{i=1}^N |P_i - \bar{x}|}{N}$$

- Number of zero values: with the amount of zero values in the residual block of 128×128 samples (V_{10}), the complexity of the prediction for that block can be estimated, since a good prediction for the first level of partitioning may mean that it is not necessary to carry out further checks at lower levels.
- Coefficient of Kurtosis (β): this defines how heavily the tails of a distribution differ from the tails of a normal distribution, depending on how the values are distributed around the average, so that a greater kurtosis implies a higher concentration of values close to the average. In this case, the kurtosis has been calculated

for the 128×128 block in both the residual frame (V_{11}) and the reconstructed image (V_{12}).

$$\beta = \frac{\sum_{i=1}^N (P_i - \bar{x})^4}{N \cdot \sigma^4} - 3$$

- Spatial index of the 128×128 block: the spatial index feature defines the level of detail in the block, that is to say, whether it is a complex region of the frame or a homogeneous zone, so it has been calculated only in the reconstructed image (V_{13}), using the Sobel filter (SF). SF is the convolution ($*$) of the Sobel matrices as indicated below, with a 3×3 matrix, A_p , surrounding the pixel to which the filter is being applied. The spatial index (SI) is calculated as the standard deviation of the value of the pixels contained in the 128×128 size block after applying the SF .

$$SF_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * A_p$$

$$SF_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * A_p$$

$$SF_p = \sqrt{SF_x^2 + SF_y^2}$$

$$SI = \sigma(SF_i)$$

- Cost in bits of the block in the HEVC stream (V_{14}).
- Number of pixels in the frame (width \times height) of the sequence to which the 128×128 block belongs (V_{15}).
- The λ value used to encode the frame (V_{16}): Since λ depends on the QP and the temporal layer of the frame in the hierarchy established by the group of pictures (GOP), it can be obtained directly from the VVC encoder.

4.3. Generation of the Dataset

Having defined the features used in the construction of the proposed decision model, it is necessary to describe the process followed to generate the instances of the dataset. Ideally, a dataset should contain instances from as many different scenarios as possible to ensure the adaptivity of the model to any context and input sequence.

The JVET published a document that defines different reference configurations used to establish a common test framework among proposals [21]. Within these configurations, the document specifies a wide set of sequences, mainly grouped by resolution, which comprise many different scenarios, such as regular footage, computer-generated content and videoconferencing. This variety ensures a wide spectrum of use cases. The list of sequences is as follows:

- Class A1 (3840×2160 pixels): Tango2, Drums100, Campfire and ToddlerFountain2.

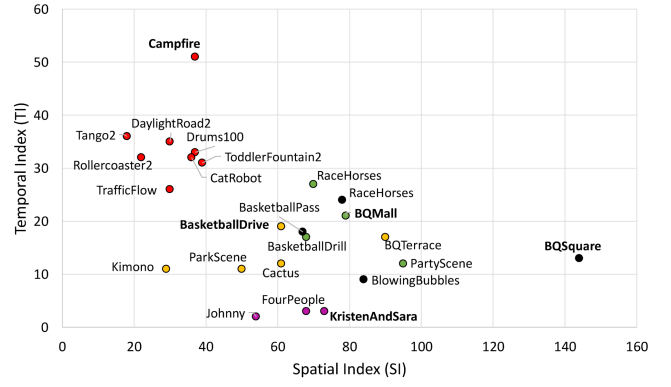


Figure 5: SI and TI of the test sequences. Classes: A (Red), B (Yellow), C (Green), D (Black) and E (Purple).

- Class A2 (3840×2160 pixels): CatRobot, TrafficFlow, DaylightRoad2 and Rollercoaster2.
- Class B (1920×1080 pixels): Kimono, ParkScene, Cactus, BQTerrace and BasketballDrive.
- Class C (832×480 pixels): RaceHorsesC, BQMall, PartyScene and BasketballDrill.
- Class D (416×240 pixels): RaceHorses, BQSquare, BlowingBubbles and BasketballPass.
- Class E (1280×720 pixels): FourPeople, Johnny and KristenAndSara.

To build our model, we selected five of the sequences for the training set. The criterion used in the selection consisted of taking one sequence per class on the basis of their spatial index (SI) and temporal index (TI) [18]. Based on Fig. 5, which shows the distribution of these indices for all sequences, we selected Campfire (Class A), BasketballDrive (Class B), BQMall (Class C), BQSquare (Class D), and KristenAndSara (Class E), given that they cover a wide range of distinctive cases. However, with the aim of homogenizing the number of instances per class and avoiding overfitting due to the significant difference in resolution, only 1,000 instances per temporal layer and sequence were selected. The instances not used for training and those corresponding to the remaining sequences were left for validation.

In addition to the type of content being transcoded, it is also important to consider configuration parameters that may affect the prediction. In particular, on the basis of the JVET document, we considered four QP values to generate the dataset, namely 22, 27, 32 and 37, which cover a wide range of rate-distortion scenarios [21]. Additionally, we considered RA configuration, which sets the GOP size to 16 frames, and the interval at which I frames are used to 1 second. It should be noted, however, that I frames were omitted from the analysis due to the new features introduced in VVC compared with HEVC, such as separate luma and chroma encodings. Therefore, only B frames were considered to build the model.

The specific HEVC implementation selected to encode and decode the streams used for feature extraction and construction of the proposed classifier was the HM version 16.16 [19]. This fully-featured HEVC reference software offers significant coding efficiency by using strategies that provide near-optimal results compared with other implementations. Regarding VVC, we selected its matching implementation, i.e., the VVC test model (VTM) version 2.0.1 [20], to perform the encoding of the HEVC streams with the aim of determining the class of the instances in the dataset. The class attribute, which represents the value to be predicted by the model, was obtained from each 128×128 block. If the block was split in four CUs using a QT, its value would be 1, and 0 otherwise.

Based on the above considerations, the total number of instances in the dataset was almost 6 million instances, of which only 1.40% were used for training the model, and the remaining 98.60% for validation and evaluation.

4.4. Construction of the Decision Model

The decision model used in the first partitioning level of the QT was generated using the WEKA software [15]. This tool, which was developed in Java, supports well-known data mining algorithms and operations such as clustering, regression and visualization. With this aim, we used the information obtained from the 128×128 blocks to generate the model.

For a better understanding of the model creation process, Fig. 6 depicts a flowchart of the different stages in the data processing, from the extraction of input information from the HEVC stream to the generation of the model. As can be seen, the first step is the characterization of each variable and attribute by its type. Different data types include numeric, nominal, string or dates. In our case, all the information extracted from the blocks is numeric in all cases except for the class attribute. The second step consists in splitting the instances in training and testing datasets as specified in the above subsection.

Once the datasets have been created, the next step is the creation of the model from the training set. Among all the possible classifiers, we selected Naïve-Bayes, which is based on the idea that an event occurs after other events that may have an influence on the former, but that are independent of each other once the class is known. Mathematically this is expressed as the factorization by the probability of the class multiplied by the probability of each variable given the class, i.e. given a class Y and a set of variables $\{X_1, \dots, X_N\}$, the following expression is satisfied:

$$P(Y|X_1, \dots, X_N) \propto P(Y) \cdot P(X_1|Y) \dots P(X_N|Y)$$

This set of frequencies is computed in only one reading, and thus the computational complexity of building a Naïve-Bayes classifier is $\mathcal{O}(Nn)$, where N is the number of instances and n the number of features [29]. In addition, Naïve-Bayes is linear in its classification phase, i.e. $\mathcal{O}(n)$, becoming one of the fastest classifiers available.

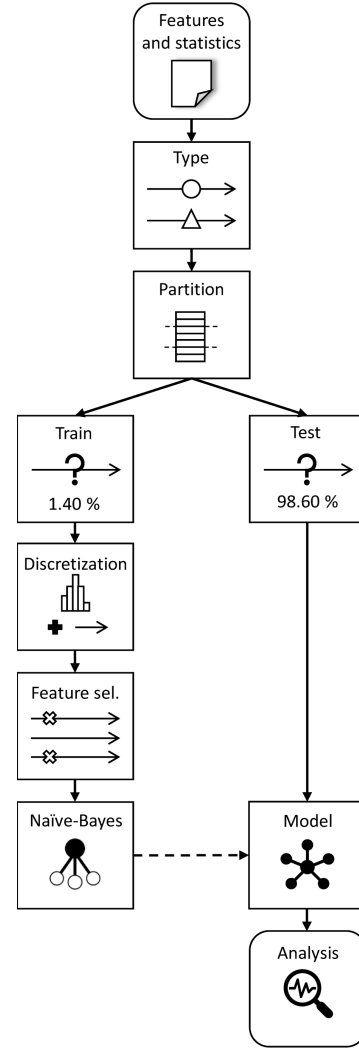


Figure 6: Data processing and model generation flowchart.

To ensure a correct classification of the model, the training dataset requires prior preprocessing, including feature discretization and selection. In this regard, it is possible to measure the performance of the classifier being constructed by using the accuracy metric defined by the following mathematical expression:

$$\text{Accuracy (\%)} = \frac{\text{True positives} + \text{True negatives}}{\text{Total number of instances}} \cdot 100$$

Without any prior preprocessing, the accuracy of the model employing a 5-fold cross validation on the training set using a Naïve-Bayes classifier is only 79.01%. Given that the input attributes are continuous quantitative variables, which forces us to assume that they follow a specific distribution, and given that Naïve-Bayes results in better accuracy with categorical variables, all attributes are discretized using a supervised discretization filter. In this way, all numerical attributes are transformed into intervals whose range depends on their contribution to the class attribute [27]. After the discretization, the accuracy of the model increases to 84.42%.

Table 2

Accuracy results of the wrapper algorithm.

Attribute	Accuracy of the classifier after the inclusion of a new attribute		
	$\{\emptyset\}$	$\{V_{14}\}$	$\{V_{14}, V_{12}\}$
$+V_1$	83.22	89.98	89.77
$+V_2$	84.39	89.99	89.89
$+V_3$	81.91	89.97	90.10
$+V_4$	84.30	89.95	89.84
$+V_5$	84.06	89.97	89.90
$+V_6$	84.40	90.00	89.91
$+V_7$	73.43	92.32	92.20
$+V_8$	78.12	92.13	92.04
$+V_9$	77.96	92.15	92.05
$+V_{10}$	84.37	89.99	89.91
$+V_{11}$	84.40	89.99	89.90
$+V_{12}$	73.51	92.34	-
$+V_{13}$	75.45	92.05	91.93
$+V_{14}$	92.33	-	-
$+V_{15}$	71.03	92.18	91.82
$+V_{16}$	72.14	91.45	91.76

Building on the preprocessing of the training set, Naïve-Bayes classifiers, like any other probabilistic classifier, are sensitive to the feature sets used to induce them. Therefore, it is necessary to discern the attributes that actually contribute to the prediction of the class variable from those that are irrelevant or redundant. This process is called feature subset selection and, among all possible selection algorithms, we selected Wrapper with forward selection to generate the corresponding training subset [13]. Forward selection is an iterative algorithm that adds variables to a set which is initially empty. To determine which attribute must be added in each iteration, the algorithm evaluates which of the remaining attributes contributes most to the accuracy of the algorithm. If none of the attributes increase the accuracy achieved by the current set of variables at a given iteration, the algorithm finishes prematurely. Considering this method evaluates only a few subsets of variables, it is computationally efficient and robust against overfitting.

Table 2 shows the evolution of the Wrapper algorithm executed on the original training set. The resulting subsets are evaluated using Naïve-Bayes classifiers and 5-fold cross-validation. On the basis of the partial results of each iteration, it can be seen that the variable selection algorithm finishes after three iterations:

- In the first iteration, all variables are tested individually. As a result, variable V_{14} , which represents the cost in bits to encode the 128×128 block in the HEVC stream, obtains the greatest accuracy, that is 92.33%.
- In the second iteration, all possible pairs formed by V_{14} and any other variable are tested. The accuracy results show that only V_{12} , which is the coefficient of kurtosis in the reconstructed block, achieves a slight increase in the accuracy of the current set, which rises to 92.34%.

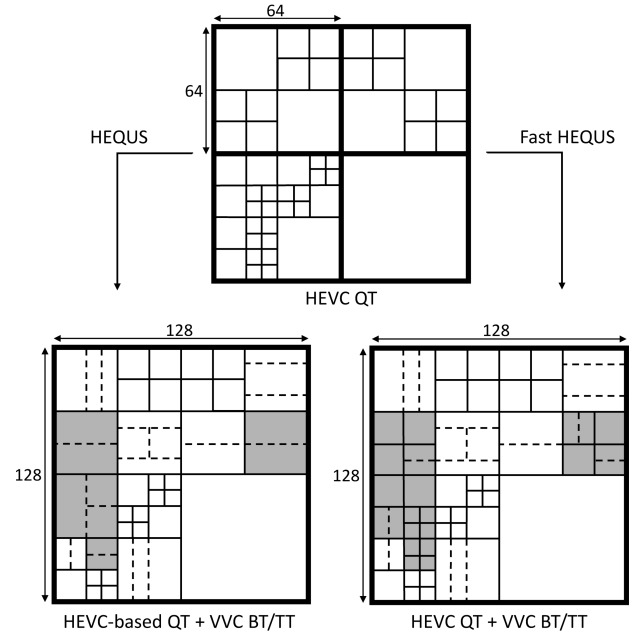


Figure 7: Partitioning example of HEQUS (left) and Fast HEQUS (right) on the basis of the original HEVC partitioning (differences shaded in gray).

- Finally, the third iteration provides no subset for which the accuracy exceeds the one of the previous iteration.

As a result of the variable selection process, only two of the variables were selected to build the model, namely V_{14} and V_{12} , omitting the remaining ones. In this regard, it is worth noting the difference in terms of accuracy of the model before and after the preprocessing of the dataset. The validation of the model using the testing set, however, will be shown as part of the evaluation of the proposal.

4.5. Partitioning Decisions in the Remaining Levels

The proposed Naïve-Bayes classifier enables the prediction of the partitioning of the first QT level in 128×128 blocks. This prediction may result in not splitting the block, in which case the QT evaluation is finished, and the encoder can either encode the block as is, or split it using BT and TT structures. By contrast, if the proposed model decides to split the 128×128 block, any rate-distortion evaluation for such a block is omitted, and the transcoder evaluates the corresponding subblocks.

The size of the blocks generated after partitioning a 128×128 block in the QT is 64×64 pixels, which matches the maximum CTU size in HEVC. In this way, and considering that the original HEVC stream went through an exhaustive RDO procedure to compress the source media, our proposal uses the existing QT structure of the input stream to encode the VVC stream. Therefore, it is assumed that the same QP is used for both input and output sequences. As mentioned previously, a set of early termination checks are performed in the VTM encoder, which can affect the brute-force scheme, since some rate-distortion evaluations and partition-

ing modes can be skipped. The behavior of these implementations is shown with shaded blocks in Fig. 7: the main difference is that the HEQUS algorithm is implemented on top of existing early QT splitting techniques, whereas the Fast HEQUS algorithm adopts the exact HEVC partitioning at 64×64 and lower levels. The two implementations presented in this proposal are detailed below:

- Implementation of the HEQUS algorithm: VTM implements several early termination techniques that allow the encoder to omit some rate-distortion evaluations and partitioning modes [6]. These techniques, which can be applied to any block type and level of partitioning, can influence the partitioning decisions of the HEQUS algorithm and its evaluation. In particular, the most important fast coding techniques are:
 - A technique that enables the caching of coding information, such as the best BT cost, the best TT cost, the best non-split cost, and the number of splits in QT, BT and TT. If the encoder needs to reuse information from other CUs, it can access the costs stored in cache.
 - A fast coding decision algorithm for CU depth called fast large CTU used to accelerate the encoding process when the maximum CTU size is set larger or equal to 128×128 pixels. In such a case, intra prediction is skipped for the current block when the area (width \times height) of the luma component is greater than 4096 samples.
 - Before evaluating a splitting decision, a skip-history rule is checked. According to this rule, if the skip mode was selected for the three past blocks in previous levels, then the block is no longer split.

Therefore, QT partitioning decisions are provided by HEVC, unless an early check prevents the same decision. In this case, the decision of the technique is taken into account, ending the QT when it is signaled. Next, BT and TT partitioning is performed without any modification with respect to the default VTM coding flow.

- Implementation of the Fast HEQUS algorithm: In this proposed transcoding method, the VTM fast checks are ignored in QT blocks, that is to say, the partitioning decisions in the transcoder are completely forced to be the same as in HEVC. Therefore, the computational time of checking more prediction modes is saved and, since the QT scheme reaches deeper levels, fewer divisions are made in the BT and TT structures, resulting in greater time savings in the transcoding process. As a result, the transcoder avoids the evaluation of the QT, and only needs to evaluate the resulting 64×64 , 32×32 , 16×16 , and 8×8 blocks along with their BT and TT structures.

In both implementations, in the case of blocks that exceed the bottom or right frame boundary, the block is forced to be split until the samples are located inside the frame boundaries. In VVC, if a portion of a block exceeds a boundary, it is forced to split in one direction using the BT structure. An implicit horizontal split is used if the block exceeds the bottom boundary, or vertical split in case of the right boundary is surpassed. Therefore, since our proposal infers only on the QT partitioning decisions, these blocks would not be affected, as they belong to the BT partitioning structure.

5. Performance Evaluation

This section presents the results obtained by the proposed HEVC-VVC transcoding approach. Firstly, the transcoding scenario and metrics used to measure its performance are presented, in terms of both computational cost and coding efficiency. Subsequently, the model in the first partitioning level is analyzed, followed by the results of the HEQUS and Fast HEQUS implementations.

5.1. Experimental Setup and Metrics

Although the model was generated using information from blocks encoded using RA configuration, the evaluation was performed with RA, low delay B (LB), and low delay P (LP) configurations [21]. The QP values used for the encoding were 22, 27, 32, and 37. Input and output sequences were encoded using 10-bit encoding and 4:2:0 chroma sub-sampling. The transcoding process followed to evaluate our approach is summarized as follows:

1. First, the original raw sequence is encoded using HM version 16.16 for each QP and configuration [19].
2. Then, each HEVC sequence is decoded back to raw format. Simultaneously, the decoder exports the statistical information used as input by the proposed model on a 128×128 block basis.
3. Once the decoded sequences and the statistical information are available, each raw video file is encoded with both the baseline transcoder and the proposed transcoding scheme. The former is an unmodified version of the VTM 2.0.1 encoder that does not make use of the statistical information, while the HEQUS algorithm is implemented on top of it [20].

Once the sequences are finally in VVC format, the results are analyzed in terms of processing time and coding efficiency by comparing both our proposal and the baseline transcoder. For this aim, we used the BD-rate and the time reduction (TR) metrics. The BD-rate is a measure of coding efficiency that represents the percentage of bitrate variation between two sequences with the same objective quality [2]. Therefore, a negative BD-rate means the obtained bitrate of our proposal is lower than the bitrate obtained with the anchor version of VTM. The TR metric, in turn, enables the assessment of time savings, and is calculated using the following expression:

Table 3

Accuracy of the proposed model.

Class	Sequence	Accuracy (%)			
		QP 22	QP 27	QP 32	QP 37
A1	Tango2	92.60	87.44	85.98	85.91
	Drums100	96.53	87.75	85.66	85.22
	Campfire	96.66	92.23	91.46	92.93
	ToddlerFountain2	99.17	97.91	97.16	94.98
A2	CatRobot	90.46	86.47	85.66	86.97
	TrafficFlow	90.07	83.43	85.26	88.90
	DaylightRoad2	95.47	85.40	81.42	85.40
	Rollercoaster2	91.60	85.90	82.97	81.66
B	Kimono	93.92	90.40	87.88	86.09
	ParkScene	94.86	86.35	83.76	84.57
	Cactus	91.90	91.40	88.92	87.77
	BasketballDrive	93.42	88.44	87.54	88.29
	BQTerrace	97.41	86.72	81.14	84.51
C	BasketballDrill	97.26	93.92	91.57	89.28
	BQMall	94.76	92.15	91.81	89.52
	PartyScene	99.24	96.60	90.56	84.74
	RaceHorsesC	99.29	98.26	97.89	96.67
D	BasketballPass	98.43	96.80	93.80	86.91
	BQSquare	99.94	93.56	80.96	84.52
	BlowingBubbles	98.50	92.23	86.43	82.07
	RaceHorses	99.66	99.20	98.51	97.24
E	FourPeople	88.23	91.97	93.20	94.20
	Johnny	88.83	90.66	92.42	95.32
	KristenAndSara	88.34	88.36	91.35	93.60
Class A1		96.24	91.33	90.06	89.76
Class A2		91.90	85.30	83.83	85.73
Class B		94.30	88.66	85.85	86.24
Class C		97.64	95.23	92.96	90.05
Class D		99.13	95.44	89.92	87.69
Class E		88.47	90.33	92.32	94.37
Average		94.86	90.98	88.89	88.64

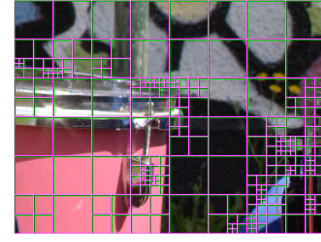
$$TR (\%) = \frac{T_{\text{reference}} - T_{\text{proposal}}}{T_{\text{reference}}} \cdot 100$$

It should be noted that the time required to obtain the statistics used by the model is included in T_{proposal} , and represents only approximately 0.1% of the total encoding time, which is negligible compared with the time saved.

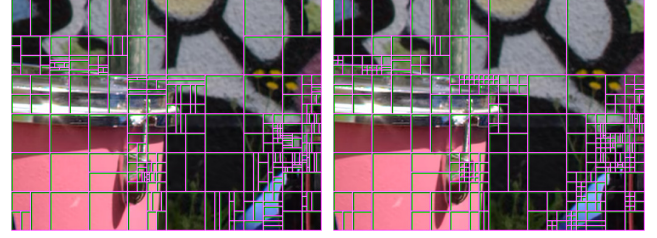
Regarding the experimental setup, the hardware platform used in the tests was composed of an Intel® Xeon® E5-2630L v3 CPU running at 1.80 GHz and 16 GB of main memory. The encoders were compiled with GCC 5.4.0-6 and executed on Ubuntu 16.04.3 LTS (GNU/Linux 4.4.0-143). Turbo Boost was disabled to achieve the reproducibility of the results.

5.2. Analysis of the Decision Model

As indicated in the previous section, the JVET defined a document with a set of sequences that should be used in



(a) Original HEVC encoding.



(b) Baseline transcoding scheme. (c) Proposed Fast HEQUS algorithm.

Figure 8: Comparison of the partitioning between the baseline transcoder and the Fast HEQUS transcoding approach.

the evaluation of proposals. The dataset used to generate the proposed model was elaborated from such sequences, and later divided into training and testing sets. While the former was used to build the decision model, the latter was left aside for its validation.

Table 3 shows the accuracy of the model using the testing set. To facilitate the understanding of the results, and to analyze the adaptivity of the model in deeper detail, the results are shown separately for each sequence and QP. It should be noted that the instances used in the training set were removed from the testing set for the Campfire, BasketballDrive, BQMall, BQSquare, and KristenAndSara sequences. In this way, the results show that the accuracy achieved is significantly higher for lower QPs, while still near 90% in the case of higher QPs. Moreover, it can be seen that there is no overfitting with respect to any QP, sequence, or class, since high accuracy is achieved in all cases.

When considering all the instances of the testing set, the average accuracy achieved by the proposed model is 89.42%. While it is slightly lower than the accuracy obtained from the training set, it is still significantly high, and thus confirms the validity of the model. In addition, the results in Table 4 show the performance of a transcoder where QT partitioning decisions at level 128×128 are given by the Naïve-Bayes model built in this proposal and, once the decision is made at the first level, the coding flow is maintained by default at the lower levels. The results related to the time reduction show that the model performs better in high-resolution classes (A1, A2, B and E), where greater time savings are achieved. For the low resolution of the test sequences belonging to classes C and D, where a block of 128×128 pixels represents a large part of the frame and, therefore, the chances of splitting this block in QT are higher, it can be seen that the time reduction is lower compared with the rest

Table 4

Performance results of the Naïve-Bayes model for the 128×128 pixel blocks over the anchor transcoder (VTM 2.0.1).

Class	Sequence	Random Access		Low Delay B		Low Delay P	
		BD-rate (%)	TR (%)	BD-rate (%)	TR (%)	BD-rate (%)	TR (%)
A1	Tango2	1.07	30.38	0.58	31.70	0.36	26.05
	Drums100	0.75	16.87	0.31	15.82	0.37	12.21
	Campfire	0.06	9.83	0.49	7.52	−0.06	6.71
	ToddlerFountain2	−0.05	6.50	−0.02	4.77	−0.04	3.87
A2	CatRobot	0.75	20.11	0.58	20.90	0.64	16.21
	TrafficFlow	0.44	24.55	1.18	30.35	0.99	24.47
	DaylightRoad2	1.35	25.52	0.11	23.40	0.27	19.20
	Rollercoaster2	0.95	30.00	0.52	31.78	−0.08	24.90
B	Kimono	0.23	17.46	0.03	14.85	0.00	11.81
	ParkScene	0.17	12.50	−0.02	9.48	0.13	8.33
	Cactus	0.08	12.33	0.10	10.46	0.27	7.72
	BasketballDrive	0.27	11.86	0.16	9.02	0.22	7.37
	BQTerrace	0.02	12.97	0.46	12.80	0.20	8.90
C	BasketballDrill	0.20	6.16	0.23	5.16	0.16	3.32
	BQMall	−0.06	6.92	0.06	4.91	0.13	4.18
	PartyScene	−0.11	4.78	0.06	3.20	0.07	2.32
	RaceHorsesC	−0.01	4.41	0.03	2.74	0.09	1.93
D	BasketballPass	0.66	3.40	0.48	2.62	0.66	1.73
	BQSquare	−0.19	6.50	−0.02	6.09	−0.15	3.86
	BlowingBubbles	0.04	4.27	0.17	3.33	0.04	2.88
	RaceHorses	0.23	3.30	0.22	2.48	0.34	2.05
E	FourPeople	0.06	15.15	0.22	15.66	0.31	12.90
	Johnny	0.33	16.36	0.32	20.52	0.55	16.28
	KristenAndSara	0.35	18.88	0.31	22.47	0.52	18.73
Class A1		0.46	15.90	0.34	14.95	0.16	12.21
Class A2		0.87	25.01	0.60	26.61	0.46	21.20
Class B		0.15	13.42	0.15	11.32	0.16	8.83
Class C		0.01	5.57	0.10	4.00	0.11	2.94
Class D		0.19	4.37	0.21	3.63	0.22	2.63
Class E		0.25	16.80	0.28	19.55	0.46	15.97
Average		0.32	13.38	0.27	13.00	0.25	10.33

of the classes, but with a negligible impact in terms of BD-rate penalty. Based on these results, we can confirm that the model applied to the first level provides a computational cost saving of around 13% with a negligible BD-rate penalty.

5.3. Experimental Results

As a first visual evaluation, Fig. 8 shows a comparison between the baseline transcoder and the Fast HEQUS algorithm transcoding scheme. This figure displays the partitioning performed in a region of the seventh frame of the *Drums100* sequence for the RA configuration and QP 27. As can be seen, both QT splittings at the 128×128 level of the tree structure are virtually identical, which highlights the accuracy of the prediction of our classification algorithm. Regarding the remaining levels, the QT structure is inherited from HEVC, whereas the new MTT structure defined in VVC is still evaluated.

Tables 5 and 6 show the results of the implemented HEQUS and Fast HEQUS algorithms, respectively, over the

anchor transcoder in terms of the BD-rate and TR for all test sequences and scenarios. As can be observed, in both cases, the RA and LP scenarios display better coding efficiency than LB. Regarding the TR, the three scenarios show equivalent time savings in each implementation, around 45% in HEQUS and 60% in Fast HEQUS. This shows the homogeneity of the model irrespective of the scenario, even when it was built from information of 128×128 blocks belonging to sequences encoded under the RA configuration.

Regarding the impact on resolutions, as seen from the individual results, Class A achieves the highest TR. This is due to the fact that sequences belonging to this class have higher resolution compared to other classes, and therefore 128×128 blocks are not split as often. In this regard, the Naïve-Bayes model can early terminate the evaluation of the QT by deciding not to split such blocks, resulting in significant time savings, in both HEQUS and Fast HEQUS implementations. As far as coding efficiency is concerned, results

Table 5

Results of the HEQUS algorithm over the anchor transcoder (VTM 2.0.1).

Class	Sequence	Random Access		Low Delay B		Low Delay P	
		BD-rate (%)	TR (%)	BD-rate (%)	TR (%)	BD-rate (%)	TR (%)
A1	Tango2	3.22	50.82	2.93	56.05	2.68	54.51
	Drums100	4.55	48.16	4.08	50.59	3.59	50.34
	Campfire	0.73	50.02	2.89	51.06	1.52	51.90
	ToddlerFountain2	0.57	46.57	0.83	47.21	0.82	47.19
A2	CatRobot	4.47	49.42	5.32	54.59	5.06	54.22
	TrafficFlow	0.97	30.10	3.20	41.48	2.88	42.70
	DaylightRoad2	2.39	52.08	2.72	55.11	2.62	54.96
	Rollercoaster2	2.05	51.68	1.62	53.08	0.99	51.95
B	Kimono	1.47	43.58	1.19	46.11	1.28	43.90
	ParkScene	2.14	39.31	2.75	44.60	2.92	41.70
	Cactus	1.95	45.74	2.35	48.42	2.25	46.51
	BasketballDrive	2.62	50.57	2.50	50.77	2.51	50.95
	BQTerrace	0.71	34.57	4.48	35.18	1.39	35.13
C	BasketballDrill	2.57	48.15	2.67	48.28	2.66	46.56
	BQMall	3.47	47.48	3.54	47.78	3.49	45.91
	PartyScene	0.96	45.14	2.04	47.33	1.49	46.22
	RaceHorsesC	2.80	50.89	2.75	52.76	2.50	52.19
D	BasketballPass	3.31	50.87	3.16	52.06	2.96	50.92
	BQSquare	0.36	35.23	2.20	42.10	0.97	38.68
	BlowingBubbles	0.91	41.25	2.34	44.80	1.80	42.81
	RaceHorses	3.72	50.05	2.82	53.15	2.98	51.02
E	FourPeople	1.58	36.88	3.45	36.08	3.19	33.78
	Johnny	1.41	26.16	4.30	23.09	4.49	18.69
	KristenAndSara	1.75	32.88	4.04	31.41	3.73	28.24
Class A1		2.27	48.89	2.68	51.23	2.15	50.99
Class A2		2.47	45.82	3.22	51.07	2.89	50.96
Class B		1.78	42.75	2.65	45.02	2.07	43.64
Class C		2.45	47.92	2.75	49.04	2.54	47.72
Class D		2.08	44.35	2.63	48.03	2.18	45.86
Class E		1.58	31.97	3.93	30.19	3.80	26.90
Average		2.11	44.07	2.92	46.38	2.53	45.04

are more constant in Classes B, C and D in the HEQUS algorithm, since this implementation is using an HEVC-based QT scheme in combination with the early termination checks of the VTM encoder. Therefore, in the Fast HEQUS algorithm, the full transfer of HEVC splitting decisions for 64×64 blocks and lower, generates a different behavior. This is explained by the fact that HEVC was mainly developed for high-definition resolutions, while the VVC standard is targeted at UHD video coding instead, including larger block sizes and more efficient tools to encode them. As a result, the HEVC partitioning may be a suboptimal partitioning for the VVC standards, which results in deviations in terms of BD-rate, as seen in CatRobot or DaylightRoad2 sequences. In spite of this, there are also cases in which the results show gains in coding efficiency, as is the case of Campfire and ToddlerFountain2 sequences.

Finally, the performance analysis of the HEQUS and Fast HEQUS algorithms shows that the latter achieves better results. In terms of coding efficiency, the BD-rate penalties

are very similar in both proposed algorithms, around 2~3%. However, since Fast HEQUS directly provides the HEVC QT partitioning ignoring VTM early checks, both the number of rate-distortion evaluations and partitioning modes are reduced, and therefore it is able to increase the TR up to 13% over HEQUS implementation, achieving time savings of around 60% on average for all sequences. This shows the partitioning decisions made by the Fast HEQUS perform better than the early skip detection decisions available in VTM.

6. Conclusions and Future Work

This paper presents a new approach for accelerating the QT partitioning decisioning of an HEVC-VVC transcoder using the proposed HEQUS algorithm. To this end, a Naïve-Bayes classifier model predicts the splitting decision of 128×128 blocks in VVC on the basis of information extracted from the HEVC stream. Given that the maximum CTU size in HEVC is 64×64 pixels, the HEVC QT parti-

Table 6

Results of the Fast HEQUS algorithm over the anchor transcoder (VTM 2.0.1).

Class	Sequence	Random Access		Low Delay B		Low Delay P	
		BD-rate (%)	TR (%)	BD-rate (%)	TR (%)	BD-rate (%)	TR (%)
A1	Tango2	3.20	69.22	3.87	73.13	2.68	72.28
	Drums100	7.82	63.53	6.69	63.66	5.44	63.14
	Campfire	-5.41	66.11	-3.99	68.07	-4.85	68.90
	ToddlerFountain2	-5.62	58.59	-2.02	59.44	-1.78	59.06
A2	CatRobot	7.39	65.26	7.90	69.49	6.74	67.95
	TrafficFlow	1.76	43.44	5.80	56.26	5.18	54.35
	DaylightRoad2	6.38	62.81	5.73	66.94	5.51	66.83
	Rollercoaster2	5.22	66.36	4.43	66.35	2.30	65.13
B	Kimono	1.66	55.43	1.58	58.03	1.20	56.26
	ParkScene	3.76	52.25	3.49	57.24	4.09	52.00
	Cactus	1.94	60.72	2.69	62.02	2.24	61.40
	BasketballDrive	3.59	66.91	3.82	65.89	3.36	66.68
	BQTerrace	-1.25	43.07	-0.58	41.73	-2.20	45.60
C	BasketballDrill	3.37	65.70	3.05	66.55	2.92	62.06
	BQMall	4.70	62.05	4.13	62.12	4.09	55.53
	PartyScene	0.20	54.59	1.28	54.65	0.42	53.19
	RaceHorsesC	3.32	63.83	2.88	67.14	2.63	64.65
D	BasketballPass	3.32	66.85	3.55	69.79	3.51	66.07
	BQSquare	-0.23	35.65	1.01	46.17	-1.37	35.68
	BlowingBubbles	-0.10	48.20	1.04	53.44	0.51	47.61
	RaceHorses	4.19	62.72	3.11	68.56	3.42	63.84
E	FourPeople	3.40	51.97	5.61	47.59	5.02	49.14
	Johnny	2.67	35.63	5.40	29.42	5.63	31.59
	KristenAndSara	2.41	49.05	5.21	45.55	4.56	48.71
Class A1		0.00	64.36	1.14	66.08	0.37	65.85
Class A2		5.19	59.47	5.97	64.76	4.93	63.57
Class B		1.94	55.68	2.20	56.98	1.74	56.39
Class C		2.90	61.54	2.84	62.62	2.52	58.86
Class D		1.80	53.36	2.18	59.49	1.52	53.30
Class E		2.83	45.55	5.41	40.85	5.07	43.15
Average		2.40	57.08	3.15	59.13	2.55	57.40

tioning has been integrated into the transcoder in two different implementations for the QT decisions for those resulting blocks where the decision of the model has been to split the 128×128 block in QT. On the one hand, the HEQUS implementation has been developed, which is combined with the early termination techniques evaluated by VTM. On the other hand, Fast HEQUS ignores these techniques and provides the transcoder with the QT partitioning, for 64×64 blocks and lower levels, directly from HEVC information.

The results evince the high accuracy of the model, achieving 89.42% accuracy on all training sequences. The performance analysis of the transcoder only with model decisions at 128×128 pixel level shows the efficiency of the Naïve-Bayes classifier, given that a TR of 13% is achieved with a negligible penalty in terms of BD-rate. Moreover, a comparison between our algorithms and the baseline transcoder shows a good trade-off in terms of coding efficiency and computational cost. In particular, the HEQUS transcoding scheme, which works in combination

with the early termination techniques available in VTM, obtains 44.07% time savings at the expense of only 2.11% BD-rate for the RA configuration, scenario in which the learning of the model was performed. The results for LB and LP configurations, in turn, show that the model is generic, achieving 46.38% and 45.04% time savings, with BD-rate penalties of 2.92% and 2.53%, respectively.

In addition, the Fast HEQUS scheme, where the QT partitioning structure of the transcoder is given entirely by the HEVC QT decisions, provides better results with time savings of 57.08%, 59.13% and 57.40%, for the RA, LB and LP scenarios, respectively, with similar BD-rate penalty.

Finally, the decisions made by our transcoder have no influence on other modules such as intra or inter prediction. In this regard, as future work, we intend to analyze the relationship between some PUs in HEVC and non-square sizes provided by the BT and TT of VVC. In addition, machine learning techniques may enable the use of information from HEVC to predict the intra directional modes in VVC.

References

- [1] AOMedia, 2019. AV1 Bitstream & Decoding Process Specification. Video Standard.
- [2] Bjøntegaard, G., 2008. Improvements of the BD-PSNR Model. Technical Report VCEG-AI11. ITU-T SG16 Q6.
- [3] Borges, A., Zatt, B., Porto, M., Correa, G., 2019. Fast HEVC-to-AV1 Transcoding Based On Coding Unit Depth Inheritance, in: 2019 IEEE International Conference on Image Processing (ICIP), pp. 3571–3575. doi:10.1109/ICIP.2019.8803482.
- [4] Bross, B., Chen, J., Liu, S., 2019. Versatile Video Coding (Draft 5). Technical Report JVET-N1001. Joint Video Experts Team (JVET).
- [5] Chen, J., Alshina, E., Sullivan, G., Ohm, J., Boyce, J., 2017. Algorithm Description of Joint Exploration Test Model 7. Technical Report JVET-G1001. Joint Video Experts Team (JVET).
- [6] Chen, J., Ye, Y., Kim, S., 2018. Algorithm Description for Versatile Video Coding and Test Model 2 (VTM 2). Technical Report JVET-K1002. Joint Video Experts Team (JVET).
- [7] CISCO, 2019. Cisco Visual Networking Index: Forecast and Trends, 2017 - 2022.
- [8] Díaz-Honrubia, A.J., Martínez, J.L., Cuenca, P., Gamez, J.A., Puerta, J.M., 2016. Adaptive Fast Quadtree Level Decision Algorithm for H.264 to HEVC Video Transcoding. IEEE Trans. Circuits Syst. Video Technol. 26, 154–168. doi:10.1109/TCSVT.2015.2473299.
- [9] Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., 1996. Advances in Knowledge Discovery and Data Mining, American Association for Artificial Intelligence, Menlo Park, CA, USA. chapter From Data Mining to Knowledge Discovery: An Overview, pp. 1–34.
- [10] Franche, J., Coulombe, S., 2018. Efficient H.264-to-HEVC Transcoding Based on Motion Propagation and Post-Order Traversal of Coding Tree Units. IEEE Trans. Circuits Syst. Video Technol. 28, 3452–3466. doi:10.1109/TCSVT.2017.2754491.
- [11] Grange, A., Alvestrand, H., 2013. A VP9 Bitstream Overview.
- [12] Grellert, M., Zatt, B., Bampi, S., da Silva Cruz, L.A., 2019. Fast Coding Unit Partition Decision for HEVC Using Support Vector Machines. IEEE Trans. Circuits Syst. Video Technol. 29, 1741–1753. doi:10.1109/TCSVT.2018.2849941.
- [13] Guyon, I., Elisseeff, A., 2003. An Introduction to Variable and Feature Selection. J. Mach. Learn. Res. 3, 1157–1182.
- [14] Ha, J.M., Bae, J.H., Sunwoo, M.H., 2016. Texture-based fast CU size decision algorithm for HEVC intra coding, in: 2016 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), pp. 702–705. doi:10.1109/APCCAS.2016.7804070.
- [15] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H., 2009. The WEKA Data Mining Software: An Update. SIGKDD Explor. Newsl. 11, 10–18. doi:10.1145/1656274.1656278.
- [16] ISO/IEC, ITU-T, 2013. High Efficiency Video Coding (HEVC). ITU-T Recommendation H.265 and ISO/IEC 23008-2.
- [17] ISO/IEC and ITU-T, 2003. Advanced Video Coding for Generic Audiovisual Services. ITU-T Recommendation H.264 and ISO/IEC 14496-10.
- [18] ITU-T, 2008. P.910 - Subjective Video Quality Assessment Methods for Multimedia Applications.
- [19] JCT-VC, 2017. HEVC Test Model Version - 16.16. "https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-16.16/".
- [20] JVET, 2018. VVC Test Model Version - 2.0.1. "https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM/-/tags/VTM-2.0.1".
- [21] Li, X., Suehring, K., 2017. Common Test Conditions and Software Reference Configurations. Technical Report JVET-H1010. Joint Video Experts Team (JVET).
- [22] Li, X., Xie, R., Song, L., Zhang, L., 2017. Machine learning based VP9-to-HEVC video transcoding, in: 2017 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), pp. 1–6. doi:10.1109/BMSB.2017.7986205.
- [23] Liapin, I., 2018. Fast H.264/H.265 to AV1 stream transcoding using a moving object tracker, in: 2018 International Symposium on Consumer Technologies (ISCT), pp. 9–13. doi:10.1109/ISCT.2018.8408927.
- [24] Ohm, J.R., Sullivan, G.J., Schwarz, H., K. Tan, T., Wiegand, T., 2012. Comparison of the Coding Efficiency of Video Coding Standards - Including High Efficiency Video Coding (HEVC). IEEE Trans. Circuits Syst. Video Technol. 22, 1669–1684. doi:10.1109/TCSVT.2012.2221192.
- [25] Shanableh, T., Peixoto, E., Izquierdo, E., 2013. MPEG-2 to HEVC Video Transcoding With Content-Based Modeling. IEEE Trans. Circuits Syst. Video Technol. 23, 1191–1196. doi:10.1109/TCSVT.2013.2241352.
- [26] Sullivan, G.J., Ohm, J.R., 2018. Meeting Report of the 12th meeting of the Joint Video Experts Team (JVET). Technical Report JVET-L1000. Joint Video Experts Team (JVET).
- [27] Usama M. Fayyad and Keki B. Irani, 1993. Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning, in: Proceedings of the International Joint Conference on Uncertainty in AI.
- [28] Vetro, A., Christopoulos, C., Sun, H., 2003. Video Transcoding Architectures and Techniques: an Overview. IEEE Signal Process. Mag. 20, 18–29. doi:10.1109/MSP.2003.1184336.
- [29] Witten, I.H., Frank, E., Hall, M.A., 2011. Data Mining: Practical Machine Learning Tools and Techniques. 3rd ed., Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.



David García-Lucas received his B.Sc. degree in Computer Science and Engineering, the M.Sc. degree in Advanced Computer Science from the University of Castilla-La Mancha, Spain, in 2016 and 2017, respectively. In 2016, he joined the Albacete Research Institute of Informatics (I3A) as a Research Assistant. He has been a visiting researcher at the Metropolitan University of Cardiff, United Kingdom. Currently, he is full-time enrolled on the PhD Program in Advanced Computing Technologies, in the Laboratory of High-Performance Networks and Architectures (RAAP) at the I3A. His research interests include multimedia standards, video transcoding and video compression.



Gabriel Cebrián-Márquez received the B.Sc. and M.Sc. degrees in Computer Science and Engineering and the Ph.D. degree in Advanced Computing Technologies from the University of Castilla-La Mancha, Spain, in 2013, 2014, and 2018, respectively. In 2013, he joined the Laboratory of High-Performance Networks and Architectures (RAAP) as a Research Assistant, Albacete Research Institute of Informatics (I3A). In 2018, he joined the Department of Computer Science, University of Oviedo, Spain, as an Assistant Professor. He has been a Visiting Researcher with Technische Universität Berlin, Germany, and Ghent University, Belgium. He has also been a Visiting Professor with Fondazione Bruno Kessler, Italy. His current research interests and areas of publication include video processing and coding, parallel and heterogeneous architectures, and 5G network technology.



Antonio J. Díaz-Honrubia received his B.Sc. degree in Computer Science and Engineering (national extraordinary award) in 2011, an M.Sc. in Advanced Computing Technologies and an M.Sc. in Computer Science and Engineering in 2012, and a Ph.D. in Advanced Computing Technologies in 2016, all of them from the University of Castilla-La Mancha. In 2017 he worked for a video distribution company in Spain, a job which he combined with a part-time professorship position at the University of Castilla-La Mancha. That same year he obtained an assistant professor position at the University of Oviedo, where he stayed for a year, after which he obtained another position at the Technical University of Madrid, where he is currently an assistant professor. His research interests include video transcoding, perceptual video coding, multimedia standards, scalable video coding, and simultaneous video coding. He has also been a visiting researcher at Ghent University (Belgium) for 4 months and the Florida Atlantic University (USA) for 3 months. He has 8 publications in these areas in international refereed journals and 11 conference proceedings.



Thanuja Mallikarachchi (S'12-M'17) received his B.Sc. (Eng.) degree with honors in Electronic and Telecommunication Engineering from University of Moratuwa, Sri Lanka in 2011. From 2011 to 2013, he was a Senior Engineer at Virtusa (pvt) Ltd., Colombo, Sri Lanka. He received his Ph.D. degree in Electronic Engineering from the Centre for Vision, Speech and Signal Processing (CVSSP) at the University of Surrey, United Kingdom, in 2017. From 2017 to 2018, he was a Research Fellow in Multimedia Communication at CVSSP. Currently, he is a Lecturer in Data Science and Informatics at the Cardiff School of Technologies, Department of Applied Computing and Engineering, Cardiff Metropolitan University, Cardiff, United Kingdom.



Pedro Cuenca received his M.Sc. degree in Physics (Electronics and Computer Science, extraordinary award) from the University of Valencia, Spain, in 1994. He received his Ph.D. degree in Computer Engineering in 1999 from the Polytechnic University of Valencia, Spain. In 1995, he joined the Department of Computer Engineering at the University of Castilla-La Mancha, Spain, where he is currently a Full Professor. He has also been a visiting researcher at Nottingham Trent University, University of Ottawa and University of Surrey. His research topics are centered on the area of video compression, QoS video transmission, and video applications for multicore and GPU architectures. He has published over 150 papers in international journals and conferences. He has served in the organization of International Conferences as Chair and Technical Program Chair. He was the Chair of the IFIP 6.8 Working Group during the 2006–2012 period. He was also the Dean of the Faculty of Computer Engineering at the University of Castilla-La Mancha from 2008 to 2016.