J. Vis. Commun. Image R. 94 (2023) 103829

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

journal homepage: www.elsevier.com/locate/jvci

Full length article

A fast full partitioning algorithm for HEVC-to-VVC video transcoding using Bayesian classifiers☆,☆☆

D. García-Lucas [a,*], G. Cebrián-Márquez [b], A.J. Díaz-Honrubia [c], T. Mallikarachchi [d], P. Cuenca [b]

[a] Department of Computer Science, University of Oviedo, Oviedo, Spain
[b] High Performance Networks and Architectures group. Universidad de Castilla-La Mancha, Spain
[c] ETS de Ingenieros Informáticos, Universidad Politécnica de Madrid, Madrid, Spain
[d] School of Technologies, Cardiff Metropolitan University, Cardiff, United Kingdom

ARTICLE INFO

Keywords:
HEVC
VVC
Transcoding
MTT
Naïve-Bayes

ABSTRACT

The Versatile Video Coding (VVC) standard was released in 2020 to replace the High Efficiency Video Coding (HEVC) standard, making it necessary to convert HEVC encoded content to VCC to exploit its compression performance, which was achieved by using a larger block size of 128 × 128 pixels, among other new coding tools. However, 80.93% of the encoding time is spent on finding a suitable block partitioning. To reduce this time, this proposal presents an HEVC-to-VVC transcoding algorithm focused on accelerating the CTU partitioning decisions. The transcoder takes different information from the input bitstream of HEVC, and feeds it to two Bayes-based models. Experimental results show a time saving in the transcoding process of 45.40%, compared with the traditional cascade transcoder. This time gain has been obtained on average for all test sequences in the Random Access scenario, at the expense of only 1.50% BD-rate.

1. Introduction

In recent years, the digital world has experienced an exponential increase in video transmitted by different sources, such as broadcast transmissions, social media and video-on-demand platforms. In addition to the growth in the amount of multimedia content being shared, this trend is accompanied by an increase in the quality of content, especially high and ultra-high definition video (HD and UHD), as well as high frame rate content. Currently, more than 75% of Internet traffic corresponds to multimedia video, which is predicted to reach 82% next year [1]. Regarding the quality of content, it is estimated that 66% of the displays will be UHD by 2023, which doubles the figure of 33% for 2018 [2]. As a result, the bandwidth to transmit video, as well as storage needs, will grow exponentially in the coming years.

To provide high-quality content to large numbers of users, video coding standards need to look for efficient video compression techniques. With this aim, and to replace the market-dominant H.264/Advanced Video Coding (AVC) [3], the Joint Collaborative Team on Video Coding (JCT-VC) published the High Efficiency Video Coding (HEVC) standard [4] in 2013. HEVC has gradually replaced AVC since it is able to double the compression rate, while the same objective video quality [5] is maintained, but at the expense of a large increase in the computational cost on the encoder side. However, as mentioned above, with the huge growth in demand for multimedia content it was to be expected that HEVC would need to be replaced by a new standard within a few years. Therefore, the international organizations ITU-T and ISO/IEC, through the Video Coding Expert Group (VCEG) and the Moving Picture Expert Group (MPEG), respectively, created the Joint Video Experts Team (JVET) in 2015. After years of work, a new standard, namely Versatile Video Coding (VVC), was released in 2020 [6].

In the development of VVC, the most promising compression techniques were tested on reference software called the VVC Test Model (VTM) [7]. Thus, VVC brings new features compared with HEVC, such as a new partitioning structure, more inter prediction modes and larger block sizes. With these new features, the compression capabilities of VVC have significantly improved with respect to those offered by HEVC for video sequences of different format and content [8]. However, achieving these compression results introduces significant costs in terms of computational complexity. As stated in [9], the computational time of VTM increases by about 300% in the random access (RA), low

☆ This paper has been recommended for acceptance by Dr Zicheng Liu.
☆☆ This paper belongs to "2.2: coding and transcoding", "2.9: standards" and "5.12: machine learning techniques" categories.
* Corresponding author.
E-mail addresses: garcialucdavid@uniovi.es (D. García-Lucas), gabriel.cebrian@uclm.es (G. Cebrián-Márquez), antoniojesus.diaz@upm.es (A.J. Díaz-Honrubia), tmallikarachchi@cardiffmet.ac.uk (T. Mallikarachchi), pedro.cuenca@uclm.es (P. Cuenca).

https://doi.org/10.1016/j.jvcir.2023.103829
Received 31 January 2022; Received in revised form 29 October 2022; Accepted 22 April 2023
Available online 28 April 2023

delay P (LP) and low delay B (LB) scenarios with respect to the HEVC reference software, called the HEVC Test Model (HM) [10]. In the all intra (AI) scenario, the encoding time increases by 1800%.

In addition to the high computational cost of VVC, another aspect that emerges when a new standard is developed is that of the conversion of multimedia content, in this case from HEVC to the new format. It is evident that many applications are interested in taking advantage of the lower bit-rate offered by the high compression of VVC. For this reason, a transcoder that converts HEVC bitstreams to VVC provides added value to content creators, services and distributors, offering interoperability between them.

In view of the above, this paper presents a fast full transcoding algorithm from HEVC to VVC. A preliminary study of the VVC encoder in which the partitioning tree of each block is defined randomly, which relates to whether the current block is evaluated or directly partitioned to the next level of depth, meaning that the traditional brute-force scheme is not applied, has allowed us to determine that 80.93% of the encoding time is spent in finding the most efficient block partitioning. With this study, we can conclude that assisting the transcoder's decision-making can achieve high time savings, so the key to this work is to find the best possible decision in order to maintain the coding efficiency, avoiding the brute-force partitioning scheme of the original coding flow. The proposal is divided into three stages in order to tackle all the partitioning levels: (i) first, a Naïve-Bayes model is applied to the new larger block size (128 × 128 pixels); (ii) then, we use HEVC bitstream decisions in the quadtree (QT) partitioning to determine the partitioning in VVC; (iii) and, finally, a second Naïve-Bayes model is used after the QT to decide whether to perform the binary tree (BT) and ternary tree (TT) partitioning of VVC.

This paper is built upon a partitioning algorithm proposed by the authors in [11], but extends it with a new classifier for BT and TT structures and a later refinement of the model through a cost analysis process in order to maximize the performance of the new classifier. The decisions taken from this analysis have achieved that only 1.57% of the prediction misses involve a compression penalty. Thanks to this extension, the experimental results have shown computational time savings in the transcoding process of 45.40% on average over the time spent by a traditional cascaded transcoder for the set of test sequences provided by the JVET and encoded in the RA scenario [12]. In terms of Bjøntegaard delta rate (BD-rate), which measures the variation in bit-rate between two sequences with the same objective video quality [5], a penalty of 1.50% is introduced.

In summary, the main novelties of this work are as follows:

1. A novel Bayesian prediction model to determine the partitioning decision of BT/TT partitioning for each leaf coding unit (CU) node based on feature set specific to the current block.
2. A novel fast HEVC-VVC transcoding algorithm that uses multiple Bayesian prediction models for each QT and BT/TT partitioning levels to expedite the CU split decisions in all levels.
3. A cost benefit analysis which can be used to trade-off the transcoding complexity to coding efficiency loss during the decision making phase.

The structure of the paper is as follows. Section 2 covers some of the most relevant related work in the literature. Section 3 includes the novelties of VVC, focusing in coding block partitioning. Then, the algorithm is detailed in Section 4, and the results of the experimental evaluation are analyzed in Section 5, including a performance comparison with state-of-the-art proposals. Finally, conclusions and future work are drawn in Section 6.

2. Related work

Heterogeneous video transcoding between different standards has been studied in order to provide interoperability between them in an efficient way, since the simple transcoder composed of a cascade process is not feasible as it would be repeating an encoding process without taking advantage of the decisions made by the previous standard, resulting in a high computational cost [13]. At the time of writing this proposal, to the best of the authors' knowledge, there are no proposals from other authors involving transcoding from HEVC to VVC.

In 2018, J.-F. Franche and S. Coulombe presented a fast algorithm for H.264-to-HEVC video transcoding [14]. This approach presents a motion propagation algorithm that creates a motion vector (MV) candidate list, where the best candidate is selected at the prediction unit (PU) block. Thanks to the estimation of the prediction error of each candidate and the use of this information in different partitions, some redundancies are avoided in the encoding process. Then, a fast mode decision method based on a post-order traversal of the CTU is introduced, including several mode reduction techniques. This proposal is 11.77× faster, with a compression penalty of 3.82%, on average respect to the cascaded pixel-domain approach.

In the same year as the above proposal, another work on transcoding between H.264 and HEVC was presented [15]. In this case, it was based on exploiting the spatial and temporal information of the bitstream coming from H.264, designing a Bayesian rule in which different elements such as motion vectors, dividing depths and bit allocations accelerate the decisions for the CU size and partitioning mode in HEVC. For the RA scenario, the authors achieved time savings of 61%, introducing a BD-rate penalty of 3.50%.

In 2019, A. Borges et al. presented a proposal for transcoding between HEVC and AV1 [16], which is the royalty-free codec designed by the AOMedia group [17]. In this work, the solution infers decisions in the encoding process of AV1 by inheriting the decoded CU size information from the HEVC bitstream. As the smallest block size allowed in AV1 is 4 × 4 pixels, a depth level map is processed to generate each 4 × 4 region of a frame in HEVC. Then, this depth level map is imported in AV1 to constrain the encoding process according to the HEVC partitioning. The proposal achieves time savings of 35.41%, and a BD-rate penalty of 4.54% is introduced.

In 2021, an algorithm based on block partitioning inheritance for VP9-to-AV1 transcoding was presented [18]. VP9, which is the encoder developed by Google, has been constantly improved over the last decade, and is considerably better than its predecessor, VP8 [19]. In this proposal, the algorithm relies on the reuse of the VP9 block partitioning during the AV1 re-encoding process. The idea is based on a statistical analysis showing the correlation between the block sizes adopted by VP9 and AV1. The authors achieve a 28% reduction in the encoding time by reducing the time taken by AV1 to find the best block partitioning. In terms of BD-rate, it results in a compression penalty of 4%.

Finally, the first HEVC-to-VVC transcoding algorithm was presented by the authors in 2021 [11], but it only tackles the QT partitioning. The algorithm first starts with a Naïve-Bayes classifier at the first depth level and then carries out the remaining QT decisions based on the QT partitioning used in HEVC. However, the BT and TT partitioning decisions are left to the VVC encoder. Despite this, when used in conjunction with the early techniques implemented in VTM, the proposed algorithm achieves time savings of 44.07% with a BD-rate penalty of 2.11%, and when the QT partitioning is used as is from HEVC, the time reduction increases up to 57%, whereas the BD-rate also increases to 2.40%.

To conclude this section, it is important to mention that the release of the VVC standard has given way to a large number of fast encoding proposals, many of them compatible with a transcoder such as the one presented in this work. Some of these proposals focus on the partitioning, such as in [20], where edge detection is used to skip vertical or horizontal partition modes in intra coding and the prediction of object motion during three frames is used to accelerate inter encoding. In this context, in [21] the acceleration of the partitioning is focused on the new TT structure of VVC, where the features extracted by

edge detection is utilized to make fast decisions about the TT splitting decision and its direction.

There has also been a growth in the use of convolutional neural networks (CNN) and other machine learning techniques for fast encoding in VVC compared to HEVC publications. In this sense, a CNN-based fast inter coding method is presented in [22], where a multi-information fusion CNN model is proposed to early terminate the CU partition by jointly using multi-domain information. There are also CNNs for specific partitioning levels, such as the one described in [23], where the 128 × 128 pixel partitioning level is targeted by a CNN to decide whether to stop the partitioning, split and continue in QT or split and continue in BT/TT partitions.

Due to the transcoding algorithms studied in the literature are specifically designed for the video coding standards involved in each work, they are not applicable to our transcoding scenario from HEVC to VVC. Therefore, the development of this VVC transcoding algorithm is opening new lines of work in this area of research.

3. Technical background

This section summarizes the new features of VVC, with respect to HEVC, as defined by the version used during the development of this proposal [24]. It should be noted that, although the latest version of the standard included some other additions, the partitioning has not been modified. This means that the algorithm is suitable for any VVC-compliant encoder implementation.

HEVC partitioning is based on CTUs of up to 64 × 64 pixels, which can be recursively divided by the QT partitioning into four CUs down to 8 × 8 pixels. The CUs are then encoded using either intra-picture or inter-picture prediction. Finally, one or more PUs can be contained in a block, which is encoded using a QT of transform units (TUs).

In VVC, the CTU size is increased to 128 × 128 pixels, and the partitioning structure is extended through the use of the multi-type tree (MTT), which allows for a more flexible and adaptive partitioning of the image. As can be seen in Fig. 1, MTT employs two stages to adapt to the local characteristics of the frame input block:

- Firstly, the QT structure is carried out, where a CU can be divided into four square-shaped blocks of equal size recursively.
- Secondly, the QT leaf nodes can be split by horizontal and vertical blocks using the BT and TT structures, except for 128 × 128 pixel QT leaf nodes, for which only the BT is allowed.

In the inter prediction module, VVC includes several new features. One of them is the VVC affine motion compensated prediction, in which two motion compensation models are used. The subblock-based temporal MV prediction (SbTMVP) tool is similar to the temporal MV prediction (TMVP) technique of HEVC, including improvements in its implementation. Also, the adaptive MV resolution (AMVR) technique allows the encoding of the MVs in units of four-luma-sample, integer-luma-sample or quarter-luma-sample. With respect to the intra prediction module, the main novelty lies in the extension to 65 prediction modes from the 33 used in HEVC, allowing VVC to adapt to the characteristics of the image, although the Planar and DC modes have not been modified. Finally, the main novelties in the remaining modules are the modifications to the transform, where a multiple transform selection (MTS) is implemented in VVC. MTS allows the encoder to choose among a set of different transform functions, namely DCT-II, DCT-VIII and DST-VII. Lastly, a high-precision MV storage is used, with up to 1/16 fraction accuracy for merge, affine and MV storage.

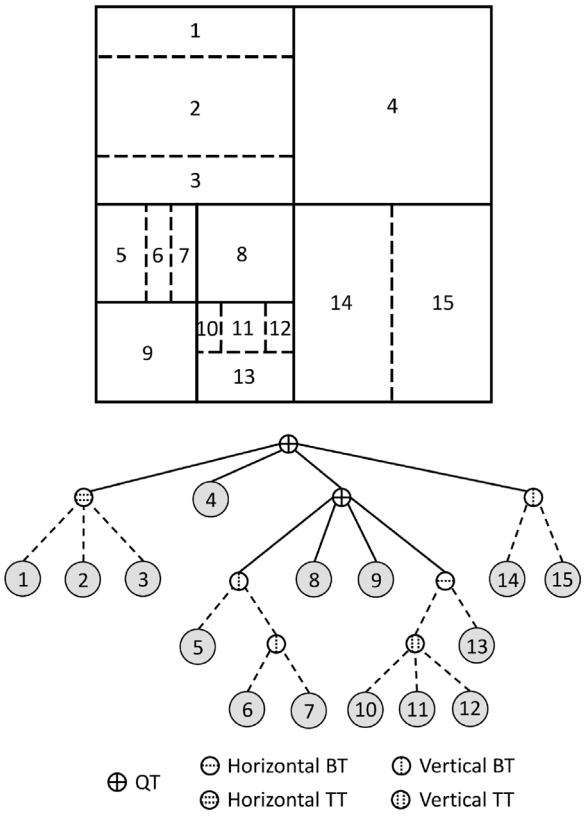


Fig. 1. MTT partition scheme of a CTU in VVC.

4. Partitioning decision algorithm

In this section, the design of the full HEVC-to-VVC transcoding algorithm is described. Its objective is to replace the exhaustive search of block partitioning, speeding up the decision-making of the transcoder at all the partitioning levels. As shown in Fig. 2, the algorithm is divided into three stages:

- Stage 1: a Naïve-Bayes classifier is applied at the 128 × 128 level to decide whether the block must be divided into 4 sub-blocks in QT, or the QT partitioning should end at the first level.
- Stage 2: if the decision is to split in QT at Stage 1, we take advantage of the HEVC bitstream to provide the QT partitioning decisions for the remaining levels.
- Stage 3: when QT partitioning is completed, a new Naïve-Bayes model is applied to decide whether or not the BT and TT structures must be evaluated.

4.1. Stage 1: Classifier applied at the block level of 128 × 128 pixels

Since VVC has been designed for ultra-high resolution, it was necessary to extend the maximum CTU size to 128 × 128 pixels, while the maximum size for HEVC was 64 × 64 pixels. In other words, there is no direct relationship at the first level of partitioning between the two standards. However, a study on the percentage of times that the 128 × 128 pixel block is not further split into CUs reveals its impact on the encoding. Table 1 shows the percentage of unsplit blocks in high-resolution video sequences according to the quantization parameter (QP) [12], in which it can be seen that most blocks finish their QT partitioning at the 128 × 128 pixel block level in low bit-rate scenarios. For this reason, the proposed transcoding algorithm makes use of a Naïve-Bayes model that predicts when the QT partitioning should end

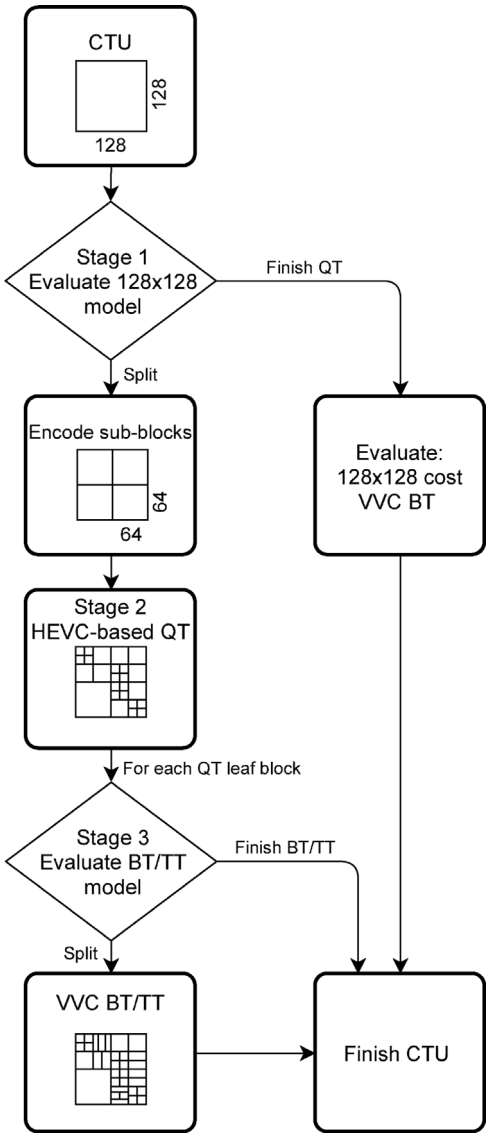


Fig. 2. Proposed HEVC-to-VVC transcoding algorithm.

at the first depth level, saving a large amount of computational time by skipping the QT at lower levels.

To develop the classifier, a large amount of valuable information from the HEVC bitstream has been analyzed following a knowledge discovery from data (KDD) approach [25]. In this process, the frames and their corresponding residual information are obtained from the HEVC bitstream, and then they are divided into 128 × 128-pixel blocks to extract information which describes the local characteristics of the blocks. Taking advantage of this information, which has been processed using machine learning techniques, a classifier that speeds up the transcoder partitioning decisions has been developed for the 128 × 128 pixel blocks. If the decision is to split, the block is divided into 4 sub-blocks of 64 × 64 pixels each. Therefore, the total computational time is reduced, since the evaluation of the current 128 × 128 pixel block and corresponding BT partitions is skipped. On the contrary, if the classifier decides not to split the 128 × 128 pixel block, the QT is skipped for the lower levels.

4.1.1. Data understanding

The features and statistics selected to develop the model for Stage 1 of the proposal are described in this subsection. This information

Table 1
Percentage of unsplit blocks per QP and class in VVC.

Sequence class	Unsplit 128 × 128 blocks (%)			
	QP22	QP27	QP32	QP37
Class A1	9.81	23.01	31.61	40.57
Class A2	15.44	32.84	46.81	59.07
Class B	10.07	25.28	39.18	53.93
Class E	48.41	65.72	75.64	82.96

describes the characteristics of the block, trying to find the correlation between the block texture and an efficient partitioning:

- Average of the block ($\overline{x}$): the complexity of the texture can be described with the average of the samples of the 128 × 128 pixel block.
- Variance of the block (σ^2): variance of the samples of a block of size 128 × 128 pixels.
- Variance of the means in sub-blocks (VM): QT partitioning divides a block into four sub-blocks of equal size. Thus, it is useful to have information about these sub-blocks by dividing a 128 × 128 pixel block into four sub-blocks of size 64 × 64 pixels. The mean of the samples of each 64 × 64 pixel block is calculated, and then the variance of these means.
- Variance of the variances in sub-blocks (VV): similar to VM, the 128 × 128 pixel block is divided into four blocks of size 64 × 64 pixels. In this case, the variance of the samples of each 64 × 64 block is calculated, and then the variance of these variances.
- Fisher coefficient of skewness (γ): the symmetry of a set of values with respect to the average can be evaluated with this metric. To calculate γ, the following expression is satisfied, where P is the value of each sample and N is the total samples of the 128 × 128 pixel block.

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

- Mean absolute deviation (MAD): this feature indicates 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:

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

- Number of zero values (Z): the complexity of the prediction for a block can be estimated with the number of zero values in its residual block of 128 × 128 pixels.
- Coefficient of kurtosis (β): this defines how sharply 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 β implies a higher concentration of values close to the average. This is the expression that satisfies:

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

- Spatial index (SI): allows to evaluate the complexity of the texture of a block using the Sobel filter (SF), so it can be determined whether it is a homogeneous region of the image. The 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 SI is calculated as the standard deviation of the pixels contained in the 128 × 128-sized block after applying the SF.

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

Table 2
Features calculated in blocks of 128 × 128 pixels.

Feature	Description
$\overline{x}$	Average of the samples of the residual block
σ^2_{Res}	Variance of the samples of the residual block
σ^2_{Rec}	Variance of the samples of the reconstructed block
VM	Variance of the means in sub-blocks of the residual block
VV	Variance of the variances in sub-blocks of the residual block
γ_{Res}	Fisher coefficient of skewness of the residual block
γ_{Rec}	Fisher coefficient of skewness of the reconstructed block
MAD_{Res}	Mean absolute deviation of the residual block
MAD_{Rec}	Mean absolute deviation of the reconstructed block
Z	Number of zero values in the residual block
β_{Res}	Kurtosis coefficient of the residual block
β_{Rec}	Kurtosis coefficient of the reconstructed block
SI	Spatial index of the reconstructed block
C	Cost in bits to encode the block in the HEVC stream
P	Number of pixels contained in a frame
λ	Lambda value used to encode the block

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

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

$$SI = \sigma(SF_i)$$

- Cost in bits of a block (C): with the amount of bits required to encode the block in the HEVC bitstream, the complexity of the prediction can be estimated.
- Number of pixels (P) in the frame (width × height) of the sequence.
- The lambda (λ) value of the frame encoding: since λ depends on the QP and the temporal layer of the frame according to the group of pictures (GOP), its value can be obtained directly from VVC.

Certain features are calculated for both the residual information and the reconstructed image of the 128 × 128 pixel block. For this reason, Table 2 provides a summary of the notation used in the document.

4.1.2. Dataset creation

To learn the Naïve-Bayes model, we start from a limited set of video sequences, specifically those that are laid out by the JVET in a common testing conditions document [12]. The main advantage of using these sequences is that they contain different scenarios, so this broad spectrum of use cases ensures that the model will be applicable to different contexts. The sequences are divided into classes according to their resolution: Class A1 (3840 × 2160 pixels), Class A2 (3840 × 2160 pixels), Class B (1920 × 1080 pixels), Class C (832 × 480 pixels), Class D (416 × 240 pixels) and Class E (1280 × 720 pixels).

Due to the limitation on the number of sequences, the dataset to train the model was created by using one sequence per class, according to their temporal index (TI) and spatial index (SI) [26], which are represented in Fig. 3. With this criterion, a wide range of textures and content is covered, where the chosen sequences are: Campfire (Class A), BasketballDrive (Class B), BQMall (Class C), BQSquare (Class D) and KristenAndSara (Class E). However, with the aim of homogenizing the number of instances per class and avoiding overfitting to the higher resolution sequences, 1000 instances per temporal layer and sequence were selected. These instances come from the encodings of each sequence for QP values 22, 27, 32 and 37, as defined in the JVET test conditions document, in frames B of the RA scenario. The instances not used for training and those corresponding to the remaining sequences, were left for the validation process.

Once the instances are obtained from the HEVC bitstream, given that this is a classification problem (i.e., supervised learning), the dataset needs to include the actual decision of the VVC encoder so that the model can learn whether the 128 × 128 pixel block should be split or not. This new attribute is called class, which is the element to be predicted by the classifier. To obtain the class, the sequences were encoded under VTM v2.0.1 [27], where the value 1 was used when a block was split in QT in VVC, and 0 otherwise.

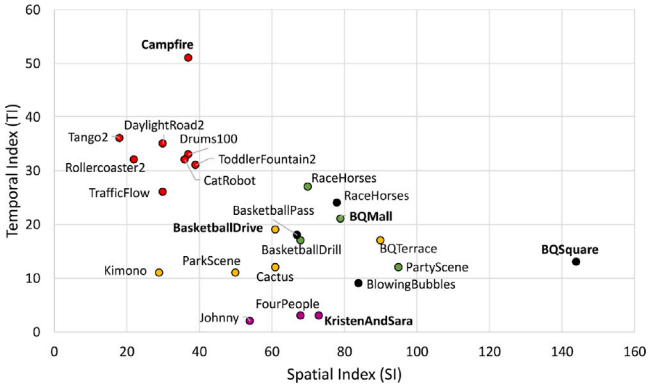


Fig. 3. SI and TI of the test sequences. Classes: A (Red), B (Yellow), C (Green), D (Black) and E (Purple).

With the selection of instances discussed above, nearly 6 million instances are available in our dataset, of which 1.40% were used in the learning phase, and the remaining (98.60%) were left for validation.

4.1.3. Learning the Naïve-Bayes model

The instances described in the previous section are used in a data process to generate the decision model in a machine learning tool called WEKA [28]. Fig. 4 depicts the different stages of this process for the creation of the Naïve-Bayes model. A Bayesian classifier was chosen because the model can be learnt in linear time, i.e., $\mathcal{O}(Nn)$, where N is the number of instances and n the number of features (which is constant and much smaller than N) [29]. In addition, it is linear in its classification stage, i.e., $\mathcal{O}(n)$, being one of the fastest classifiers available (it should be noted that n is constant, since the number of features is known). Bayes classifiers aim to classify an event (class Y) from other events that are independent of each other (variables $\{X_1, \ldots, X_N\}$), satisfying the expression:

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

To measure the performance of the model as it progresses through the stages shown in Fig. 4, an accuracy metric was used that satisfies the following expression:

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

Without preprocessing, the classifier would obtain an accuracy of only 79.01% over the training set. However, the input features are continuous quantitative variables, while Naïve-Bayes achieves better results for categorical variables. For this reason, the attributes are grouped into intervals whose range varies according to their contribution to the class attribute by a process of supervised discretization [30], increasing the accuracy of the model up to 84.42%.

The next step is to eliminate redundant information or information that does not provide information about the class. To do this, a forward selection was carried out using the Wrapper method [31]. This method consists of an iterative process that starts with an empty set, and in each iteration the variable that contributes most to the class is added, until reaching an iteration that does not improve the results of the previous iteration. When applying this method to our dataset, the Wrapper algorithm stops at the third iteration, since the set formed by the features C and β_{Rec} in the second iteration obtains an accuracy

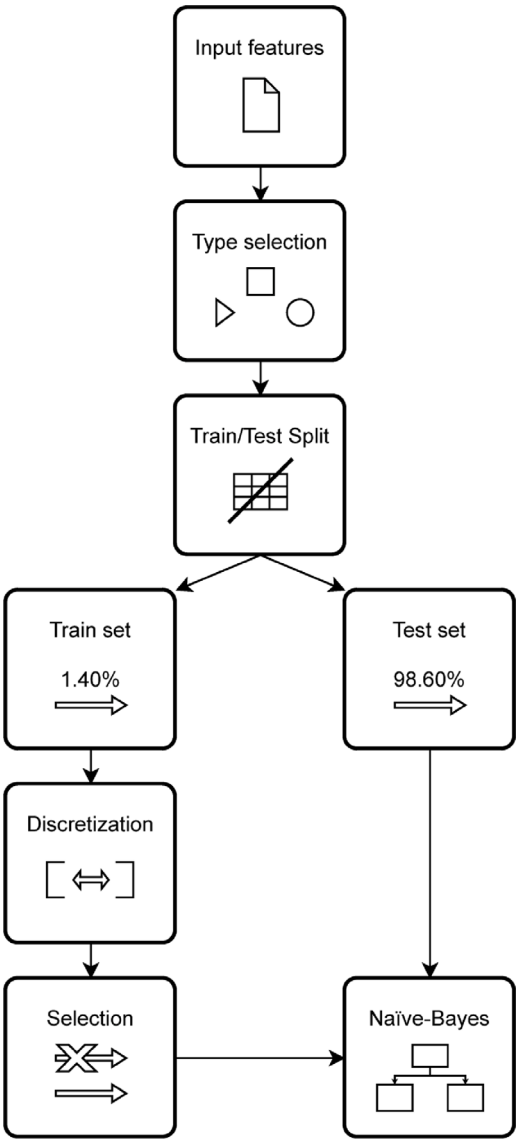


Fig. 4. Data processing and model generation flowchart.

Table 3
Accuracy per sequence class and QP of the Naïve-Bayes classifier at the 128 × 128 pixel level.

Sequence class	Accuracy (%)			
	QP22	QP 27	QP 32	QP 37
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

of 92.34% and there is no subset of three variables that improves it, finishing the algorithm.

The process performed demonstrates the importance of preprocessing the input dataset. To confirm the performance of the classifier, we evaluated it on all the remaining instances, i.e., the test set. Table 3 shows the results according to the sequence class and QP in order to verify that the model performs well in all situations, meaning that there is no overfitting to any sequence class. Thus, it can be seen that an accuracy of around 90% is achieved in all situations, with slightly better results in low QP scenarios.

4.2. Stage 2: HEVC-assisted QT decisions

In Stage 1, the proposed model decides whether or not to split the first partitioning level in QT, i.e., the blocks of size 128 × 128 pixels. If the classifier predicts not to split, the QT structure is finished and the current block can be encoded at this level or split using the BT structure. However, if the decision is to split, VVC would continue an exhaustive search for square blocks from 64 × 64 to 8 × 8 pixels. Taking advantage of the fact that these partitioning levels were already evaluated in HEVC, we can save computational time by adopting the decisions that HEVC already made in our transcoder.

Regarding the implementation of this stage in the transcoding approach, fast checks performed by VVC on the square blocks were disabled, in order to replicate the same structure in QT of HEVC from 64 × 64 pixel blocks to lower levels. These early termination techniques are useful in a VVC encoding environment. However, in a transcoding scenario, where these blocks were already analyzed in HEVC through an exhaustive rate–distortion optimization procedure, we can take advantage of this analysis by ignoring these early termination techniques, leaving fewer partitioning levels to be evaluated in BT and TT.

Once the HEVC square block structure is replicated, the QT partitioning ends, giving way to the evaluation of the block into horizontal and vertical sub-blocks using the new BT and TT partitions of VVC. In the next stage, a new model that decides whether or not it is efficient to implement these partitioning structures is introduced.

4.3. Stage 3: Classifier applied to the QT leaf blocks

The BT and TT structures of VVC make use of horizontal and vertical partitions to adapt to the local characteristics of the image more accurately. A Bayesian model was implemented to save time by skipping the BT and TT schemes. This model decides whether to finish partitioning at the current block level or to continue evaluating the horizontal and vertical partitioning blocks of the BT and TT structures. However, since the blocks are smaller compared with the classifier used in Stage 1, the number of features was increased with respect to the model of Stage 1 in order to try to capture the block details. Subsequently, a cost analysis was performed to minimize the impact of errors in this model. The following subsections explain the process carried out in this stage.

4.3.1. Data understanding

With respect to the model applied at the 128 × 128 pixel level, the amount of statistical information obtained for each instance was substantially increased. It should be taken into account that, in addition to the current block, it is necessary to analyze the features of the two horizontal blocks in BT, the two vertical blocks in BT, the three horizontal blocks in TT and the three vertical blocks in TT, making a total of 11 features for the following statistics: $\overline{x}$, σ^2_{Rec}, γ_{Rec}, MAD_{Rec}, β_{Rec} and SI. The remaining features used are:

- Dimensions (width and height) of the block.
- Temporal layer of the frame in which the current block is contained.
- QT depth of the block.
- P and λ, in the same way as the Stage 1 model.
- Variance of means and variance of variances of the samples by sub-blocks. For example, in horizontal BT, the mean of the samples of the upper horizontal sub-block and the mean of the samples of the lower horizontal sub-block are obtained, and the variance of these values is calculated. The same is done for all other possible partitionings and for the variance.

Table 4
Distribution of instances (%) in the initial confusion matrix of the Stage 3 classifier.

		Predicted	
		Skip BT/TT	Check BT/TT
Actual	Skip BT/TT	75.53%	6.67%
	Check BT/TT	5.49%	12.31%

- Since Stage 2 ends with the corresponding HEVC CU in QT, we obtained information about this block from the HEVC bitstream. This information corresponds to the size of the PUs, whether its encoding was inter or intra, and the number of bytes required for its encoding. It is foreseeable that this information will be very useful in the new model. On the one hand, a 2N×2N PU may indicate that no further partitioning of the block is necessary. On the other hand, the cost of encoding the current block in HEVC was one of the features chosen in the model of Stage 1.

4.3.2. Model generation

The process carried out to generate the model is similar to the process carried out for Stage 1 (see Fig. 4), with a subsequent cost analysis. Using the same sequences to obtain the training set instances, we obtained the features described in the previous subsection for 1000 instances per temporal layer, QT depth and sequence, each sequence being encoded using the QP values 22, 27, 32 and 37.

Once the feature discretization and selection had been performed, the Wrapper algorithm selected a subset of 11 variables: width and height of the block, QT depth, P, λ, γ_{Rec} of the middle block in horizontal TT partitioning and of the right block in vertical TT partitioning, β_{Rec} of the middle block in vertical TT partitioning, the variance of the variances of the samples in horizontal TT partitioning, PU partitioning of the CU in HEVC, and the cost in bytes required to encode the block in HEVC.

With these variables, the Naïve-Bayes classifier is able to achieve an accuracy of 87.84%. However, when looking at the confusion matrix shown in Table 4, we can see that the errors are homogeneously distributed (around 6%) in each of the two possible prediction errors. The impact of these two errors is not the same in terms of coding efficiency and encoding time. A false negative, that is, deciding to skip the BT and TT because of a prediction error has a high penalty in compression, since we are stopping a square block from being divided horizontally or vertically to adapt to the characteristics of the image, causing a costly block prediction. On the contrary, a false positive means checking BT and TT when it would not have been necessary because the compression efficiency is not improved. For this reason, the next subsection details the cost analysis performed to tune the decision model.

4.3.3. Cost analysis of the classifier

In order for this model to have the least possible impact on compression, it is necessary to increase the threshold of not splitting the QT leaf block by BT and TT. In this way, we allow the new BT and TT structures to be skipped in the appropriate blocks. For this purpose, two modifications were made to the model, described in the next paragraphs.

Firstly, a cost analysis was developed to minimize the impact of the false negatives. This analysis shows that the normalized probability of the decision to skip BT and TT partitioning must be greater than the 0.82 threshold to obtain the lowest impact on this prediction error, as shown in Fig. 5. In this sense, the cost refers to the BD-rate penalty introduced by the classifier, while the benefit is related to the time saved in decision-making with respect to the traditional brute-force scheme.

However, this modification implies an increase in false positives where BT and TT partitioning is predicted to be checked when it should not be evaluated, as can be seen in Table 5 with the new classifier confusion matrix. This problem was solved at the time of algorithm implementation, allowing that when the model decision is to check BT and TT, the cost of the current block is also evaluated. Thus, if that cost is greater than not using BT or TT splitting, the same result as in the case of deciding to skip them will be obtained.

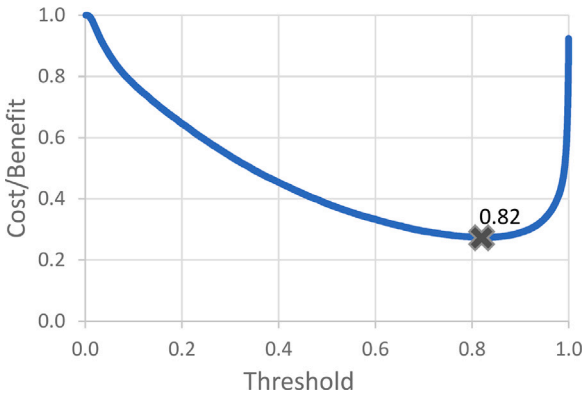


Fig. 5. Cost/Benefit of decision to skip BT/TT partitioning based on the threshold value.

Table 5
Distribution of instances (%) in the final confusion matrix of the Stage 3 classifier.

		Predicted	
		Skip BT/TT	Check BT/TT
Actual	Skip BT/TT	65.76%	16.44%
	Check BT/TT	1.57%	16.23%

In summary, the classifier will perform a correct prediction in 65.76% of the blocks, resulting in time savings, and will only produce a compression penalty in 1.57% of the cases. In the remaining 32.67% blocks, there will be neither coding time savings nor penalization, since, as the established threshold is not exceeded, the usual VVC encoding flow is maintained.

5. Performance evaluation

This section presents the results of the proposal, previously introducing the procedure to evaluate its performance and the testing environment in which the simulations were carried out.

5.1. Transcoding procedure and setup

We used the video sequences specified in the common test conditions document by the JVET [12]. Each of these sequences was encoded using 10 bits per sample encoding and 4:2:0 chroma subsampling for QP values 22, 27, 32 and 37, in the RA, LB and LP scenarios.

First, the sequences are encoded and decoded in HEVC (HM v16.16 [10]). During the decoding process, the information necessary for the three stages of the proposed algorithm is extracted, while obtaining the sequences in raw format. Then, the sequences are encoded in VVC (VTM v2.0.1 [27]) in order to extract the partitioning decisions for the classifiers. Finally, the transcoding algorithm has been implemented in VVC (VTM 17.0 [7]). Therefore, the video sequences have been encoded in the original VTM 17.0 encoder and with the proposal in order to obtain the results in terms of coding efficiency and acceleration of the transcoding process.

The tests were carried out on a hardware platform consisting of 37 nodes with an Intel® Xeon® E5-2630L v3 CPU and 16 GB of main memory, running at 1.80 GHz with Turbo Boost disabled to ensure reproducibility.

Table 6
Time saved in the transcoding process of the proposed HEVC-to-VVC transcoder.

Class	Sequence	Random access		Low delay B		Low delay P	
		BD-rate (%)	TR (%)	BD-rate (%)	TR (%)	BD-rate (%)	TR (%)
A1	Tango2	2.52	45.52	3.22	49.73	3.30	47.73
	Drums100	3.62	45.18	4.31	47.61	4.74	46.92
	Campfire	2.38	53.45	2.83	51.91	3.15	50.95
	ToddlerFountain2	0.79	37.13	0.62	39.66	0.45	38.97
A2	CatRobot	3.85	49.10	6.62	52.35	6.37	51.24
	TrafficFlow	0.28	32.18	3.16	39.47	3.51	38.60
	DaylightRoad2	1.77	50.25	3.25	54.45	3.83	52.64
	Rollercoaster2	2.27	46.45	2.47	45.36	2.14	43.39
B	Kimono	1.15	39.50	1.79	44.94	1.91	45.18
	ParkScene	1.10	45.60	3.00	54.10	3.16	51.30
	Cactus	1.48	47.19	3.00	53.06	2.60	51.45
	BasketballDrive	1.99	50.85	2.32	51.60	2.41	50.85
	BQTerrace	−1.49	37.96	2.11	44.28	0.76	46.54
C	BasketballDrill	1.99	51.14	2.30	51.37	2.28	49.54
	BQMall	2.80	49.94	3.77	52.59	3.47	50.03
	PartyScene	−0.01	52.08	1.40	54.32	0.86	50.95
	RaceHorsesC	2.16	55.46	1.79	55.45	1.48	53.49
D	BasketballPass	2.04	52.95	2.53	52.90	2.36	49.32
	BQSquare	−1.29	42.09	1.66	50.27	0.18	45.68
	BlowingBubbles	−0.08	47.89	1.75	52.79	1.39	49.47
	RaceHorses	2.40	55.30	2.66	55.88	2.31	51.84
E	FourPeople	1.42	37.80	4.17	46.34	4.05	44.67
	Johnny	1.18	29.87	4.90	37.34	5.15	35.60
	KristenAndSara	1.55	34.72	4.78	40.26	4.89	39.08
Class A1		2.33	45.32	2.75	47.23	2.91	46.14
Class A2		2.04	44.50	3.88	47.91	3.96	46.47
Class B		0.85	44.22	2.44	49.60	2.17	49.06
Class C		1.74	52.16	2.32	53.43	2.02	51.00
Class D		0.77	49.56	2.15	52.96	1.56	49.08
Class E		1.38	34.13	4.62	41.31	4.70	39.78
Average		1.50	45.40	2.93	49.08	2.78	47.31

5.2. Experimental results

Table 6 shows the time savings with respect to the MTT partitioning and the BD-rate results achieved by the proposed algorithm for each sequence and class. These results are shown for the RA, LB and LP scenarios. It is important to note that the overhead introduced by the algorithm is less than 0.1%, which is included in the times shown in Table 6, so we can conclude that the impact of the algorithm is negligible compared with the time savings achieved by the proposal.

Regarding the coding efficiency results in the RA scenario, it can be seen that we introduce a penalty of only 1.50% on average in terms of BD-rate, which is a small penalty compared with the high computational cost of 45.40% that is reduced in the transcoding process compared with a traditional cascaded transcoder. Two main conclusions can be drawn from these results. On the one hand, the Naïve-Bayes models applied in Stages 1 and 3 are effective, where prediction errors have a minor impact. On the other hand, Stage 2 takes the decisions in QT of HEVC, producing a different partitioning than VVC would obtain with the new coding tools.

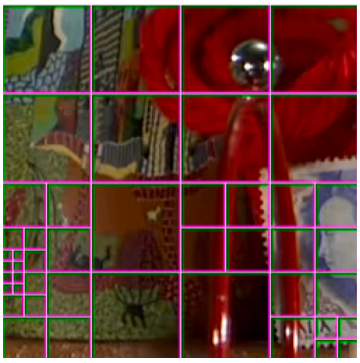
We can also observe some cases in which the transcoder obtains a better encoding efficiency. This is due to the fact that throughout the development of VTM, new coding tools have been implemented that can alter the partitioning tree. Since these techniques are ignored to respect the decision of the classifiers in Stages 1 and 3, and of HEVC for the case of QT partitioning in Stage 2, a different partitioning tree is generated than that of the baseline transcoder, which shows that VTM does not always get the most optimal compression decision. In the same way, the results for the LB and LP scenarios are similar in terms of time savings compared with the RA configuration.

With respect to time savings, we can see that the values are similar between sequences and classes around the average. Thus, we can conclude that the algorithm works properly in different scenarios, with no overfitting in its development. On the contrary, as far as coding efficiency is concerned, it can be seen that the penalty introduced is more diverse between sequences with respect to the average per class to which they belong. Given that the models developed using the Naïve-Bayes classifier reached an accuracy of 92.34% in Stage 1, and in Stage 3 only 1.57% of the cases entail penalties, we can deduce that the main impact on BD-rate is introduced in Stage 2. Therefore, we can conclude that the HEVC QT partitioning is a suboptimal solution when applied to VVC, producing more or less effective results which may even produce gains in compression, as in the case of the BQTerrace, PartyScene, BQSquare and BlowingBubbles video sequences.

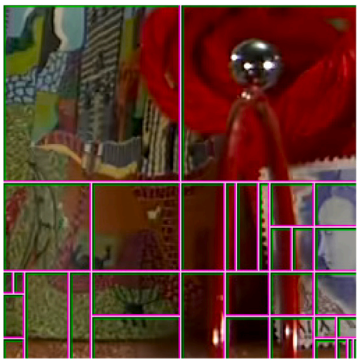
To understand how the algorithm works, Fig. 6 shows a visual comparison of the partitioning of the different schemes involved. For this purpose, the partitioning of original HEVC encoded stream, cascaded HEVC-VVC transcoded stream (baseline), and the transcoded stream generated by the proposed algorithm have been represented, showing a portion of the second frame of the Cactus sequence, encoded with QP 32 in the RA scenario. If we focus on the first partitioning level, we can see that HEVC starts its partitioning with 64 × 64 pixel blocks, while VVC starts with 128 × 128 pixel blocks. If we compare Figs. 6(a) and 6(b) at this level, we can see that the model applied in Stage 1 succeeded in its prediction, since the proposed algorithm has decided not to split in the two blocks of 128 × 128 pixels in the upper part of the image, and has decided to split in those of the lower part. When the model decides to split a 128 × 128 pixel block, Stage 2 of the algorithm is applied, performing the HEVC QT partitioning, as can be seen in Fig. 6(c), which matches the HEVC partitioning. Finally, the new Bayesian classifier is applied to decide when to continue with the BT and TT partitions durante the Stage 3 of the proposal.

5.3. Comparison with state-of-the-art fast encoding methods

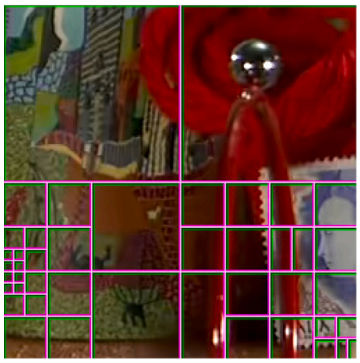
As discussed in Section 2, implementing fast encoding algorithms can also be used to expedite the transcoders. This could be done in



(a) Original HEVC encoding.



(b) Baseline VVC transcoder.



(c) Proposed VVC algorithm.

Fig. 6. Comparison between the partitioning in HEVC, the baseline transcoder and the proposal.

two ways: by directly combining the transcoder described in this work with state-of-the-art proposals that affect other encoder modules or, in the case of those proposals that influence the partitioning decisions, by working together to refine the behavior of the classifiers. For this reason, the time reductions of the fast encoding algorithms are comparable with the transcoder. However, BD-rate penalties cannot be compared as easily, given that fast encoding algorithms are typically evaluated using raw sequences as input, whereas the efficiency of transcoding proposals is assessed using already encoded material. Since the re-encoding of a video sequence introduces higher BD-rate losses, it is possible to assume that fast encoding algorithms will result in greater BD-rate penalties in a transcoding scenario compared to fast encoding.

Table 7 presents the TR of the proposal compared with some state-of-the-art works described in Section 2, in the RA scenario. This table shows the time savings achieved per sequence, as well as the average. To offer a fair comparison of these works with our proposal, Table 7 shows at the bottom the average achieved by our proposal when taking into consideration only the set of sequences in common with each related work.

In general, when comparing individual sequences, the acceleration obtained in this proposal is superior in most cases. An important detail to note is that although [20–22] have similar time reductions on average, [22] uses a CNN, reporting an algorithm overhead of 2.14% in VTM 6.0. This is a relevant cost with respect to the 0.1% required by the Bayesian transcoder classifiers applied in Stage 1 and Stage 3 of our proposal. Regarding [23], authors achieve a 9.33% time reduction by using a CNN at the 128 × 128 pixel level, so it is foreseeable that the extension of the proposal to more levels will accelerate even more. However, if we compare this proposal with the time saving obtained by our Bayesian classifier of Stage 1, i.e., only targetizing the 128 × 128 pixel level, we obtain a time reduction of 13.38%, as shown in [11].

Other recent approaches are [32,33], both implemented on top of recent versions of VTM, achieving time savings of 27.85% and 24.42%, respectively for the RA configuration. However, since these proposals do not include individual results of each sequence, they were not considered in the comparison carried out in this section.

6. Conclusions and future work

This paper describes a full transcoding algorithm for partitioning decisions applied in an HEVC-to-VVC transcoder. The proposal is divided into three stages. Stage 1 consists in a Naïve-Bayes model applied at the 128 × 128 pixel block level. If this block is split, the remaining QT levels are taken based on the HEVC bitstream, corresponding to Stage 2 of the algorithm. Finally, Stage 3 employs a new Bayesian model to decide whether or not to apply the BT and TT structures at the QT leaf nodes. The cost analysis performed on this classifier has substantially optimized the performance of this classifier.

The results show time savings in the transcoding process of around 45%–50% on average for all the sequences and scenarios evaluated, making it one of the fastest transcoders in the literature on heterogeneous video transcoding, at the expense of a BD-rate penalty of 1.50% in the RA scenario, and 2.93% and 2.78% for LB and LP configurations, respectively. Since this proposal only involves partitioning decisions, we intend to accelerate other modules of the transcoder as future work, such as the prediction of intra directional modes. In addition, the use of machine learning techniques is increasing in the field of video encoding due to the low impact on computational cost. Thus, different compatible models could be applied, leading to a joint decision which could further help in handling the coding efficiency of the transcoding process.

Table 7
Time savings (%) of the proposal compared to state-of-the-art methods (RA scenario).

Class	Sequence	Proposed	N. Tang [20]	Z. Pan [22]	Y. Ciou [21]	W. Yeo [23]
A1	Tango2	45.52	—	38.56	—	25.95
	Drums100	45.18	—	—	—	—
	Campfire	53.45	—	38.23	44.61	7.36
	ToddlerFountain2	37.13	—	—	—	—
A2	CatRobot	49.10	—	36.84	—	25.68
	TrafficFlow	32.18	—	—	—	—
	DaylightRoad2	50.25	—	35.47	—	—
	Rollercoaster2	46.45	—	—	—	—
B	Kimono	39.50	41.82	—	—	—
	ParkScene	45.60	31.60	—	—	—
	Cactus	47.19	33.17	29.36	28.27	13.80
	BasketballDrive	50.85	42.15	37.28	—	15.65
	BQTerrace	37.96	29.47	20.21	—	13.54
C	BasketballDrill	51.14	28.73	29.23	40.31	7.43
	BQMall	49.94	33.27	27.48	—	3.99
	PartyScene	52.08	35.23	20.80	24.63	2.88
	RaceHorsesC	55.46	33.89	26.39	32.11	3.16
D	BasketballPass	52.95	24.33	26.97	34.31	3.37
	BQSquare	42.09	23.00	14.86	15.13	3.54
	BlowingBubbles	47.89	21.87	22.15	17.86	3.59
	RaceHorses	55.30	31.83	24.20	31.93	0.70
E	FourPeople	37.80	26.65	33.77	17.99	—
	Johnny	29.87	24.44	35.22	—	—
	KristenAndSara	34.72	25.32	36.50	16.79	—
Average		45.40	30.42	29.64	27.63	9.33
Average TR (%) of the proposal using the same set of sequences as the related works						
	N. Tang [20] sequence set → 45.65 Z. Pan [22] sequence set → 46.86 Y. Ciou [21] sequence set → 48.19 W. Yeo [23] sequence set → 46.06					

Data availability

Data will be made available on request

Acknowledgments

This work has been supported by the Regional Government of Castilla-La Mancha under project SBPLY/21/180501/000195, by the Spanish Ministry of Science, Innovation and Universities and the European Commission (FEDER) under projects PID2021-123627OB-C52 and PID2021-128167OA-I00, and by the Spanish Ministry of Education, Culture and Sports under Grant FPU16/05692.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] CISCO, Cisco visual networking index: Forecast and trends, 2017 - 2022, 2019.
[2] CISCO, Cisco annual internet report (2018–2023) white paper, 2020.
[3] ISO/IEC, ITU-T, Advanced Video Coding for Generic Audiovisual Services. ITU-T Recommendation H.264 and ISO/IEC 14496-10, 2003.
[4] ISO/IEC, ITU-T, High Efficiency Video Coding (HEVC). ITU-T Recommendation H.265 and ISO/IEC 23008-2, 2013.
[5] G. Bjøntegaard, Improvements of the BD-PSNR Model, Technical Report VCEG-AI11, ITU-T SG16 Q6, 2008.
[6] B. Bross, J. Chen, S. Liu, Y.-K. Wang, Versatile Video Coding (Draft 10), Technical Report JVET-S2001, Joint Video Experts Team (JVET), 2020.
[7] JVET, Vvc test model version - 17, 2022, https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM/-/tags/VTM-17.0.
[8] Y.-K. Wang, R. Skupin, M.M. Hannuksela, S. Deshpande, Hendry, V. Drugeon, R. Sjöberg, B. Choi, V. Seregin, Y. Sanchez, J.M. Boyce, W. Wan, G.J. Sullivan, The high-level syntax of the versatile video coding (VVC) standard, IEEE Trans. Circuits Syst. Video Technol. (2021).
[9] G.J. Sullivan, J.-R. Ohm, Meeting Report of the 12th Meeting of the Joint Video Experts Team (JVET), Technical Report JVET-L1000, Joint Video Experts Team (JVET), 2018.
[10] JCT-VC, HEVC Test Model Version - 16.16, 2017, https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-16.16/.
[11] D. García-Lucas, G. Cebrián-Márquez, A. Díaz-Honrubia, T. Mallikarachchi, P. Cuenca, Cost-efficient HEVC-based quadtree splitting (HEQUS) for VVC Video Transcoding, Signal Process., Image Commun. 94 (2021) 116199.
[12] X. Li, K. Suehring, Common Test Conditions and Software Reference Configurations, Technical Report JVET-H1010, Joint Video Experts Team (JVET), 2017.
[13] A. Vetro, C. Christopoulos, H. Sun, Video transcoding architectures and techniques: an overview, IEEE Signal Process. Mag. 20 (2) (2003) 18–29.
[14] J. Franche, S. Coulombe, 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 (12) (2018) 3452–3466.
[15] Y. Zhang, J. Zha, H. Chao, Fast H.264/AVC to HEVC transcoding based on compressed domain information, in: 2018 Data Compression Conference, 2018, pp. 207–216, http://dx.doi.org/10.1109/DCC.2018.00029.
[16] A. Borges, B. Zatt, M. Porto, G. Correa, Fast HEVC-to-AV1 transcoding based on coding unit depth inheritance, in: 2019 IEEE International Conference on Image Processing, ICIP, 2019, pp. 3571–3575, http://dx.doi.org/10.1109/ICIP.2019.8803482.
[17] AOMedia, AV1 bitstream & decoding process specification, 2019, Video Standard.
[18] A. Borges, D. Palomino, B. Zatt, M. Porto, G. Correa, Fast VP9-to-AV1 transcoding based on block partitioning inheritance, in: European Signal Processing Conference, EUSIPCO, 2021, pp. 555–559, http://dx.doi.org/10.23919/Eusipco47968.2020.9287453.
[19] D. Mukherjee, J. Han, J. Bankoski, R. Bultje, A. Grange, J. Koleszar, P. Wilkins, Y. Xu, A technical overview of VP9-the latest open-source video codec, SMPTE Motion Imaging J. 124 (2015) 44–54.
[20] N. Tang, J. Cao, F. Liang, J. Wang, H. Liu, X. Wang, X. Du, Fast CTU partition decision algorithm for VVC intra and inter coding, in: 2019 IEEE Asia Pacific Conference on Circuits and Systems, APCCAS, 2019, pp. 361–364, http://dx.doi.org/10.1109/APCCAS47518.2019.8953076.
[21] Y.-S. Ciou, M.-J. Chen, C.-H. Yeh, C.-R. Lu, M.-C. Hsieh, C. Lo, Fast multi-type tree partition for H.266/VVC inter coding, in: 2022 IEEE International Conference on Consumer Electronics - Taiwan, 2022, pp. 471–472, http://dx.doi.org/10.1109/ICCE-Taiwan55306.2022.9869026.
[22] Z. Pan, P. Zhang, B. Peng, N. Ling, J. Lei, A CNN-based fast inter coding method for VVC, IEEE Signal Process. Lett. 28 (2021) 1260–1264.

[23] W. Yeo, B.-G. Kim, CNN-based fast split mode decision algorithm for versatile video coding (VVC) inter prediction, J. Multimed. Inf. Syst. 8 (2021) 147–158.
[24] B. Bross, J. Chen, S. Liu, Versatile Video Coding (Draft 5), Technical Report JVET-N1001, Joint Video Experts Team (JVET), 2019.
[25] U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, Advances in knowledge discovery and data mining, in: From Data Mining To Knowledge Discovery: An Overview, American Association for Artificial Intelligence, Menlo Park, CA, USA, 1996, pp. 1–34.
[26] ITU-T, P.910 - Subjective video quality assessment methods for multimedia applications, 2008.
[27] JVET, VVC Test Model Version - 2.0.1, 2018, https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftwareVTM/tags/VTM-2.0.1.
[28] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I.H. Witten, The WEKA data mining software: An update, SIGKDD Explor. Newsl. 11 (1) (2009) 10–18.
[29] I.H. Witten, E. Frank, M.A. Hall, Data Mining: Practical Machine Learning Tools and Techniques, third ed., Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2011.
[30] U.M. Fayyad, K.B. Irani, Multi-interval discretization of continuous-valued attributes for classification learning, in: Proceedings of the International Joint Conference on Uncertainty in AI, 1993.
[31] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, in: L. Kaelbling (Ed.), J. Mach. Learn. Res. 3 (2003) 1157–1182.
[32] Y. Liu, M. Abdoli, T. Guionnet, C. Guillemot, A. Roumy, Light-weight CNN-based VVC inter partitioning acceleration, in: 2022 IEEE 14th Image, Video, and Multidimensional Signal Processing Workshop, IVMSP, 2022, pp. 1–5, http://dx.doi.org/10.1109/IVMSP54334.2022.9816276.
[33] Z. Liu, H. Qian, M. Zhang, A fast multi-tree partition algorithm based on spatial-temporal correlation for VVC, in: 2022 Data Compression Conference, DCC, 2022, p. 468, http://dx.doi.org/10.1109/DCC52660.2022.00079.