Noname manuscript No.
(will be inserted by the editor)

Accelerating the CU Partitioning Decision in an HEVC-JEM Transcoder

D. García-Lucas[1*], G. Cebrián-Márquez[2], A. J. Díaz-Honrubia[3], and Pedro Cuenca[1]

Received: date / Accepted: date

Abstract High Efficiency Video Coding (HEVC) is currently the latest video coding standard available on the market, and it is able to offer up to twice the coding efficiency, in the range of 50% bitrate reduction for the same video quality, of the previous standard, namely H.264/Advanced Video Coding (AVC). HEVC was standardized in 2013 for videos up to a resolution of 2K. However, the popularity of 4K videos is increasing due to the growing use of video-on-demand platforms. Therefore, the ITU-T Video Coding Expert Group (VCEG) and the ISO/IEC Moving Picture Expert Group (MPEG) created the Joint Video Exploration Team (JVET) in 2015 to design the future video coding technology under the Joint Exploration Model (JEM), which its latest version achieves an improvement in coding efficiency of 30%, but at a high cost in terms of computational complexity (10×) with respect to HEVC. The new video standard is expected to be ready in 2020, so it is necessary to find efficient mechanisms to convert current content to the new format adopted in JEM. In this regard, our proposal consists in a probabilistic classifier based on Naïve-Bayes that enables the prediction of the splitting decision at the first quadtree level in JEM, reducing the computational complexity of the transcoding process from HEVC to this new standard. The experimental results show a good trade-off between coding efficiency and complexity compared with the

This work has been supported by the MINECO and European Commission (FEDER funds) under project TIN2015-66972-C5-2-R, by the JCCM under the project SBPLY/17/180501/000353 and by the Spanish Education, Culture and Sports Ministry under grant FPU 16/05692.

[1] High Performance Networks and Architectures Laboratory. University of Castilla-La Mancha, Spain. E-mail: {David.GarciaLucas, Pedro.Cuenca}@uclm.es
[2] Computer Science Department.
University of Oviedo, Spain. E-mail: CebrianGabriel@uniovi.es
[3] A. J. Díaz-Honrubia is with the Universidad Politécnica de Madrid, ETS de Ingenieros Informáticos. E-mail: AntonioJesus.Diaz@upm.es
* Corresponding Author

anchor transcoder, obtaining a time reduction up to 12.71% at the expense of low coding efficiency penalties in the configurations evaluated.

Keywords HEVC · H.265 · JEM · Transcoding · CTU Splitting

1 Introduction

Among all the fields of industry, the multimedia sector is one of the most demanded ones. In fact, the consumption of multimedia contents has grown exponentially in the past few years. Nowadays, 70% of total Internet traffic is video traffic, and it is forecasted to reach 82% in 2020 [1]. Year after year, users are demanding higher video qualities, larger resolutions and new formats in order to enrich their viewing experience. In this scenario, video coding standards play a critical role in managing the huge bandwidth and storage requirements of these contents, as well as in regulating all these emerging formats.

H.265/High Efficiency Video Coding (HEVC) [2] was developed in 2013 by the Joint Collaborative Team on Video Coding (JCT-VC) to replace the H.264/Advanced Video Coding (AVC) standard [3], which has been the most widely used codec in recent years in many applications, such as broadcasting, multimedia, streaming and telephony systems. HEVC roughly doubles the compression performance of H.264/AVC, especially for high definition (HD) and ultra-high definition (UHD) content, but at a cost of extremely high computational complexities in the video encoding process [4].

In spite of the superior performance of HEVC, the most recent user demands introduce new challenges that require even more efficient compression techniques. With this in mind, 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 Exploration Team (JVET) to analyze the potential need for the standardization of future video coding technologies with a compression capability that significantly surpasses the one achieved by HEVC, especially for streaming UHD, panorama video content from sports events, concerts, shows, 360° omnidirectional immersive multimedia and high-dynamic-range (HDR) video content. Since the creation of the JVET, the most promising future technologies explored have been integrated into the Joint Exploration Test Model (JEM) [5].

JEM has been the experimental software of the JVET, and contains coding tools that have been designed to achieve a coding performance superior to that of HEVC. The latest version of JEM achieves an improvement in coding efficiency of nearly 30% with respect to HEVC, but at the cost of an extremely high computational complexity (10×). For this reason, it is necessary to develop more efficient compression techniques. The official standardization activities for the next video coding standard beyond HEVC started in April 2018 after evaluating the submissions to the Call for Proposals (CfP) for future video coding technologies [6]. The next video coding standard is currently

expected to be finalized by the end of 2020, and will be named the Versatile Video Coding (VVC) standard [7]. The new standard is estimated to enable the delivery of UHD services at bitrates that today are used to carry HD television signals. In addition, VVC will enable twice as much video content to be stored on a server or be sent via a streaming service.

In this scenario, in which multiple standards coexist, video transcoding has come a long way. Video transcoding is the term used to refer to the digital conversion of data. The process of video transcoding is normally a two-step process. The first part of the process is decoding, in which the original data is converted to an uncompressed format. The second part is re-encoding, whereby the data is transformed to the format used by the destination device, usually different from the original one. The need for transcoding originated from the rapid change in digital media and the increasing demand for new video formats.

Considering both the superior compression performance expected from JEM and the large amount of existing content that is currently encoded using the HEVC standard, a transcoder that converts bitstreams from HEVC to JEM will be of great value to many applications, taking advantage of the superior performance offered by JEM to provide interoperability between the HEVC standard and the format of the future video compression standard. Furthermore, HEVC encoders are widely available on the market and provide a good trade-off in terms of rate-distortion (RD) and cost. Hence, providing a cost-effective encoding method is mandatory to address the issue of the lack of dedicated JEM encoders. The union of an HEVC encoder with an efficient HEVC-to-JEM transcoder might be the solution to this problem, offering the benefits of the superior RD performance of the JEM standard while giving the new JEM-compliant devices the possibility of processing contents encoded with JEM.

However, all the new coding tools integrated into JEM involve a considerable increase in terms of encoding time. Among them, one of the tools that contributes most to the improved coding efficiency of JEM is its new partitioning scheme. This scheme features a maximum block size of 128×128 pixels, while in HEVC the maximum size is 64×64 pixels. Therefore, there is no direct relationship between a block of 128×128 pixels in JEM and any block in HEVC. For this reason, this work introduces a probabilistic model based on the Naïve-Bayes (NB) algorithm to analyze statistical information of the source stream with the aim of accelerating the splitting decision of 128×128 pixel blocks. As a result, the computational complexity involved in the conversion of contents from HEVC to JEM is reduced. The experimental results show that the proposed algorithm, compared with the anchor transcoder in the configurations evaluated, can achieve time reductions of up to 12.71% on average over the full set of the JEM common test sequences, with a penalty lower than 1.35% in terms of the Bjøntegaard delta rate (BD-rate) [8], which measures the increment in bitrate while maintaining the same objective quality.

This paper is organized as follows. Section 3 highlights key features of the JEM coding design, focusing on the partitioning structure. Section 2 includes relevant related work. The proposed model is described in Section 4,

and an analysis of the results of both the model and the implementation of the proposal on the JEM reference software is presented in Section 5. Finally, Section 6 concludes the paper.

2 Related Work

The conversion of video content between standards (heterogeneous transcoding) has been studied in some depth in recent years. The simplest transcoding process performs the complete process of decoding and fully re-encoding [9], but this is not time effective. The proposals available in the literature try to avoid unnecessary operations at the encoding stage, or even to accelerate complex modules by using information collected in the decoding process as part of the transcoder. In this section, several works in the literature that focus directly on video transcoding between standards will be analyzed.

A proposal focused on MPEG-2/H.264 heterogeneous transcoding is presented by Fernandez-Escribano et al. in [10]. It describes a decision algorithm for the prediction in P frames at the macroblock (MB) level using machine learning techniques. Using these techniques, decision trees are constructed to classify the information of the MPEG-2 MBs in one of the H.264 coding modes. The proposed algorithm only requires the average and variance of the residue in MPEG-2 in order to implement a simple decision tree based on the quantization paramenter (QP) value selected in the encoding stage in H.264/AVC. The results show that the proposed algorithm is able to maintain a good picture quality while reducing the computational complexity by as much as 95%, with a negligible impact on the quality of the transcoded video.

In 2012, regarding H.264/HEVC heterogeneous transcoding, Peixoto and Izquierdo [11] proposed the reuse of motion vectors as well as a similarity metric to decide which coding unit (CU) partitions should be tested for HEVC. This proposal obtains a maximum speed-up of 4.13× with a BD-rate penalty of up to 10.92%. One year later, Peixoto et al. [12] proposed two alternatives to map H.264/AVC MBs into HEVC CUs based on a machine learning (ML) model: one of them is an off-line model and the other uses an on-line training stage. An extension of these two works was published in [13], in which the first k frames of the sequence are used to compute the parameters, so that the transcoder can learn the mapping for that particular sequence. Then, two different types of mode mapping algorithms are proposed. In the first solution, a single H.264/AVC coding parameter is used to determine the outgoing HEVC partitions using dynamic thresholding. The second solution uses linear discriminant functions to map the incoming H.264/AVC coding parameters to the outgoing HEVC partitions. The first solution obtains a trade-off between the speed-up and bitrate increase of 3.08× and 16.2%, respectively.

Jiang et al. [14] proposed a transcoder algorithm based on region feature analysis in 2014. The main idea consists in dividing each frame into three regions in terms of coding tree units (CTUs) on the basis of the correlation between image complexity and the coding bits of the H.264/AVC source bit-

stream. The results obtained in terms of speed-up and BD-rate are 1.93× and 1.73% on average, respectively.

In [15], a complete transcoding algorithm between the standards H.264/AVC and HEVC is presented by Diaz-Honrubia et al. in 2016. The probabilistic model developed in this proposal is based on Naïve-Bayes for each level of partitioning (64×64 and 32×32) and temporal layer, making a total of 8 models. In addition, each model is constructed with the information of 26 variables extracted from the decoder of H.264/AVC. These variables are calculated for 1000 instances for each of the 4 sequences trained and QP values (22, 27, 32, and 37). The full implementation of the transcoding algorithm achieves 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.

Finally, 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 [16]. The motion propagation algorithm creates a motion vector candidate list at CTU level, and then selects the best candidate at prediction unit (PU) level. This method avoids computational redundancy by precomputing the prediction error of each candidate at CTU level, and by reusing the information for various partition sizes. 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 this paper, we present a first soft computing approach to video transcoding between HEVC and JEM. To date, there are no known proposals that involve the conversion of contents from HEVC to JEM, so a new line of research is opened in the field of heterogeneous video transcoders.

3 Technical Background

As mentioned above, JEM is the new test model software under study by the JVET group, and it has been built on top of the HEVC test model (HM) [17]. The basic encoding and decoding flowchart of HEVC is kept unchanged in JEM. However, the design elements of the most important modules, such as the modules of block structure, intra and inter prediction, residue transform and loop filter, are modified and new coding tools are added. This section includes some technical background to the new standard, describing the most important features of the said modules [18].

In HEVC, the image partitioning is defined by CTUs, with a maximum size of 64×64 pixels, that are split into CUs by using a quadtree structure to adapt to various local characteristics. The decision whether to code a pic-

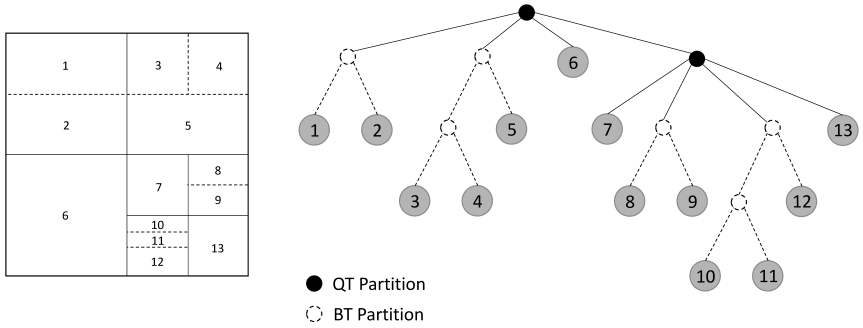


Fig. 1 Example of a QTBT structure.

ture area using inter-picture (temporal) or intra-picture (spatial) prediction is made at the CU level. Each CU can contain one or more PUs, according to the PU splitting type, and transform units (TUs), according to another quadtree structure similar to the coding tree for the CU. In JEM, this concept of multiple partition types, including CU, PU and TU, is no longer needed with the incorporation of the quadtree plus binary tree (QTBT) structure for blocks [19, 20]. This provides more flexibility for CU partition shapes to better match the local characteristics of the video sequence.

In this block structure, CTUs have a maximum size of 256×256 pixels, although this is limited to 128×128 pixels in JEM. Each CTU is first partitioned by a quadtree structure into square CUs. Then, leaf nodes can be further partitioned by a binary tree structure. By the use of this tree, each CU can be split into horizontal and vertical CUs. An example of a CTU structure with its associated QTBT in JEM is depicted in Figure 1. First, the CTU is split into four blocks of equal size by the use of a quadtree, and once the leaf nodes are reached, the binary tree begins the horizontal and vertical divisions.

For intra prediction, in order to capture finer edge directions present in natural videos, the directional intra modes are extended from 33, as defined in HEVC, to 65. The Planar and DC modes remain the same. These new directional prediction modes are applied for all block sizes, in both luma and chroma components. The additional directional modes are depicted as blue arrows in Figure 2, where the existing HEVC modes are shown with black arrows. Other additional intra features include cross-component linear model (CCLM) and new interpolation filters.

Regarding inter prediction, with QTBT a new concept of sub-CU appears, and this comes from the technique used to improve the motion information by splitting a large CU into sub-CUs and deriving motion information for all the sub-CUs of the large CU. Each sub-CU contains motion information, which can be obtained using either the alternative temporal motion vector prediction (ATMVP) or the spatial-temporal motion vector prediction (STMVP) techniques. Additionally, the accuracy for the internal motion vector storage and the Merge candidate increases to 1/16 samples. Moreover, JEM incorporates the overlapped block motion compensation (OBMC) technique, which had al-

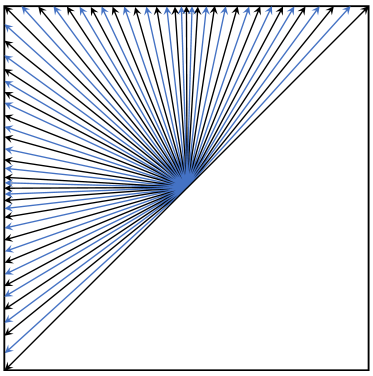


Fig. 2 Intra prediction modes in JEM.

ready been implemented in previous standards. Finally, JEM also includes certain frame-rate up conversion (FRUC) techniques, which allows to derive motion information on the decoder side.

Regarding the modifications to the transform, JEM incorporates some new functions, and an adaptive multiple transform (AMT) that allows the encoder to choose among a large set of transform functions to encode the intra and inter CU residual information.

As far as the in-loop filtering is concerned, in addition to the deblocking filter and the sample adaptive offset (SAO) operation applied in HEVC, JEM introduces two new filters, namely the bilateral filter and the adaptive loop filter (ALF). The bilateral filter is the first loop filter in the decoding process, just after a block is reconstructed. ALF, in turn, is applied last, following a block-based filter adaption approach.

4 Proposed HEVC-JEM Transcoding Model

The transcoding approach proposed in this paper uses a probabilistic model that helps the transcoder make the decision of splitting the CU block under study at the first depth level of the quadtree included in the QTBT. The decoding information in HEVC will be exploited in a model based on Naïve-Bayes classifiers in order to assist the quadtree decision on CU splitting in JEM, at the 128×128 pixel level. The use of this kind of classifiers is due to their flexibility, simplicity and strong independence assumption: all input features are conditionally independent between them given the class in the generation of the output decision.

4.1 Description of the Proposal

By the use of a knowledge discovery from data (KDD) process [21], some useful information in the form of statistics about the HEVC video stream can be

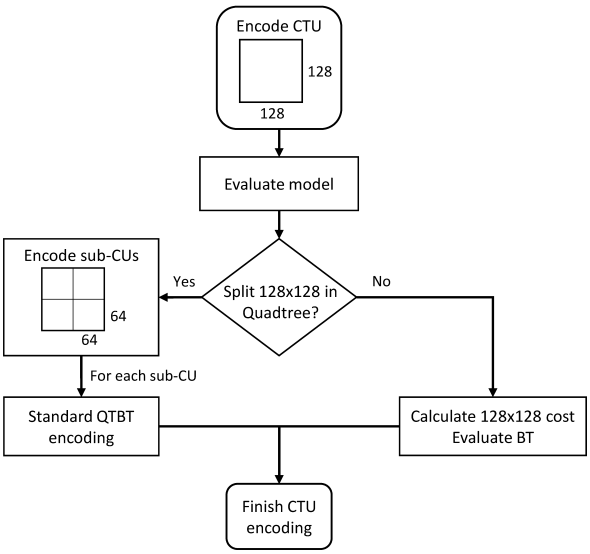


Fig. 3 Encoding process of the proposal in the JEM encoder.

obtained in its decoder. Then, this information is preprocessed and later processed using ML techniques to convert it into a mathematical model that can be executed in the on-line transcoding process. In other words, our proposal replaces the brute force scheme used in the implementation of JEM with a low-complexity algorithm based on a NB classifier. This approach is based on the idea that there is no direct relationship between a CU of 128×128 pixels in JEM and any block in HEVC (maximum size of 64×64 pixels). Therefore, our motivation is to analyze statistical information from the input frames, dividing them into blocks of 128×128 pixels, and to create a decision model that saves computing time by deciding whether to split or not the 128×128 blocks in JEM.

The model created from this information is incorporated into the coding flow of the JEM encoder, where the encoding process has been modified to integrate the proposal into the QTBT structure. The new coding process is shown in Figure 3. This diagram shows the effect of integrating the model into the coding process, achieving a reduction in coding time regardless of whether the partitioning decision is to split the first level (128×128 pixels) or not. On the one hand, if the decision of the model is to split, the encoder saves the computation time of the first level, in both QT and BT, dividing the block into 4 CUs of 64×64 pixels each. On the other hand, if the model decides not to split the block in the first level, the encoder saves the computation time of checking the QT and BT of lower levels, since it only evaluates the QT and BT of the first level.

To design the model, the steps described in the following subsections were carried out, such as choosing the sequences to extract the input information,

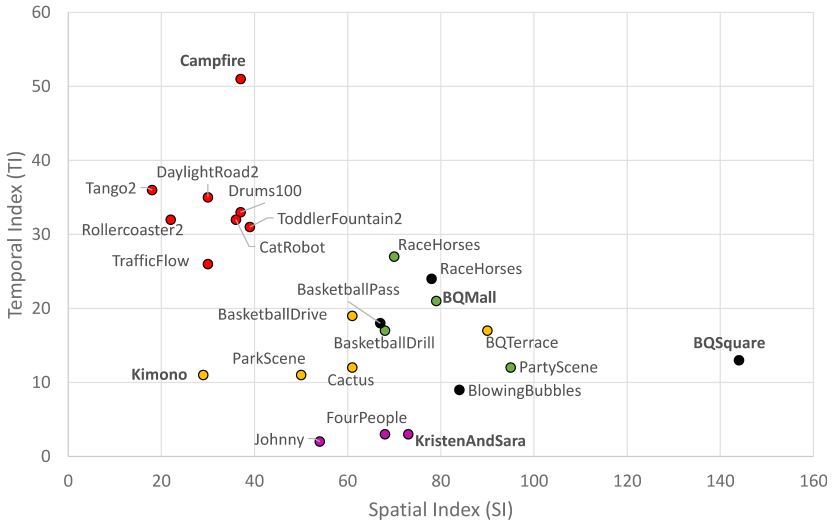


Fig. 4 SI and TI of the test sequences. Classes: A (Red), B (Yellow), C (Green), D (Black) and E (Purple).

selecting the variables that will be obtained from these sequences, and generating the model with the WEKA software [22].

4.2 Selection of Sequences

The JVET group issued a document with information about the video sequences that should be used in the evaluation of proposals implemented on JEM, as well as the reference configuration that should be used in the encoding process [23]. On the basis of this document, the criterion for choosing the sequences from which the information will be extracted consisted in selecting one sequence per class (corresponding to different resolutions) based on its spatial index (SI) and temporal index (TI), according to the ITU-T P.910 recommendation [24].

The SI index and the TI index were obtained for all the test sequences. A graphical distribution of these indices is shown in Figure 4. In order to cover a wide range of distinctive features, the selected sequences were: Campfire (Class A), Kimono (Class B), BQMall (Class C), BQSquare (Class D) and KristenAndSara (Class E).

4.3 Variables and Statistical Information Analyzed

For the development of the decision model based on HEVC information, a large number of variables were selected that could define the behavior of the quadtree at the 128×128 pixel level. Later, during the feature selection process,

some of them will be discarded if considered redundant or irrelevant for the model.

This set of information was obtained in complete blocks of size 128×128 in the available B frames of each of the selected sequences mentioned in the previous section, for the QP values 22, 27, 32 and 37. Each feature is described by its general expression, and the nomenclature V_1-V_{16} is used to represent the variables for the corresponding block in the residual frame and in the reconstructed image, depending on the information that is being calculated. Thus, the initial set of features contains the following variables:

- Average of the block ($\overline{X}$): average of the samples in the 128×128 residual block (V_1). The following expression shows the calculation of $\overline{X}$, where P is the residue value of each sample and N is the total of samples contained in a block of size 128x128:

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

- 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).

$$\sigma^2 = \frac{\sum_{i=1}^{N} (P_i - \overline{x})^2}{N}$$

- Variance of the means in sub-blocks: the 128×128 residual block is divided into 4 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: the 128×128 residual block is divided into 4 blocks of size 64×64. The 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): this coefficient allows the evaluation of the skewness of a set of values based on their distribution around the average. This statistic has been calculated for the 128×128 block in both the residual frame (V_5) and the reconstructed image (V_7).

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

- Mean absolute deviation (MAD): this makes it possible to obtain the variation in a set of values by calculating the average distance between each value and the average. This statistic has been calculated for the 128×128 block in both the residual frame (V_6) and the reconstructed image (V_8).

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

- Number of zero values: number of zero values in the 128×128 block of the residue (V_{10}).

– Coefficient of Kurtosis (β): this measures how values are grouped around the average, so that greater kurtosis implies a higher concentration of values close to the average. This statistic has been calculated for the 128×128 block in both the residual frame (V_{11}) and the reconstructed image (V_{12}).

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

– Spatial index of the 128×128 block, obtained only in the reconstructed image (V_{13}), using the Sobel Filter (SF), which is the convolution ($*$) of the Sobel matrices as indicated bellow, with a 3×3 matrix, A_p, surrounding the pixel to which the filter is being applied). This can indicate whether it is a block with many details or, on the contrary, that it is a homogeneous region. 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.

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

$$SF_p = \sqrt{SF_x^2 + SF_y^2} \qquad SI = \sigma(SF_i)$$

– Cost in bits to encode the block in the compressed HEVC stream (V_{14}).
– Number of pixels in the frame (width × height) of the sequence to which the 128×128 block belongs (V_{15}).
– Lambda value used to encode the frame (V_{16}). This depends on the QP value and the position of the frame within the GOP pattern.

The variables described above were calculated for the sequences selected in the previous section. These sequences were previously encoded and later decoded with version 16.16 of HM [17], and using the reference coding parameters described in the document of common evaluation conditions [23].

4.4 Learning the Model in WEKA

After the previous step, we have the information of all the 128×128 blocks available in the B frames of the selected sequences for each QP. First of all, the types of all variables are assigned (e.g., categorical, integer, real, etc.). Then, due to the different resolution and number of frames of the sequences, the criterion of selecting up to 1000 instances per temporal layer, sequence and QP has been adopted to form the train set (what leaves the 98.61% of the samples for the test set), avoiding an overfit to the information of blocks belonging to higher resolution sequences. In addition, the class attribute is defined for each instance in the training set, which represents the value that the model should predict from the statistics and variables defined in the previous section. This attribute indicates whether or not the block of size 128×128 is

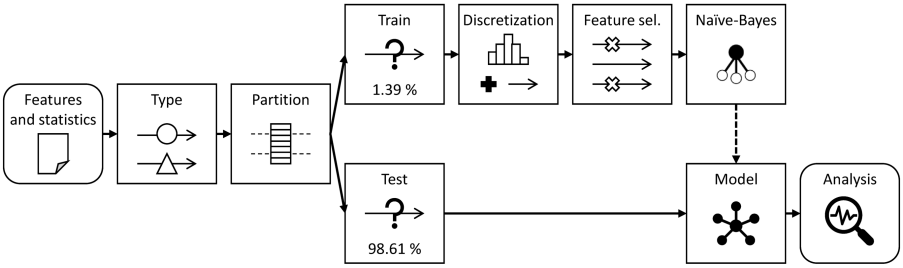


Fig. 5 Data processing and model generation with WEKA.

split into 4 blocks of size 64×64, and it has been obtained by encoding and decoding the sequences in JEM 7.1 [5].

Once the dataset is ready, the process of creating and learning the probabilistic model starts with WEKA [22] as shown in Figure 5, which is a tool developed in Java that supports the most popular data mining algorithms and tasks, such as clustering, regression and visualization.

To measure the performance of the model at the different stages described below, the accuracy metric is used, which represents the total percentage of correctly classified instances:

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

The accuracy obtained by using a 10-fold cross validation on the training set with the Naïve-Bayes classifier before any preprocessing is only 79.54%. Therefore, the first step carried out in WEKA is the preprocessing of data through the discretization of variables. Most of the features are continuous quantitative variables, which forces us to assume that they follow a specific distribution when working with probabilistic models. The discretization of variables allows the generation of a new dataset in which the variables become categorical, whereby the information is grouped in intervals with similar information. The supervised discretization algorithm generates a set of intervals for each variable, these having a different rank depending on the information with which each interval contributes to the class [25]. Once the variables are discretized, the accuracy of the model including all statistics is increased to 88.85%.

Probabilistic classifiers in general, and those based on Naïve-Bayes in particular, are quite sensitive to the feature set used to induce the classifier. Thus, the presence in the training set of irrelevant or redundant variables may significantly affect the precision of the learned classifier. Because of this, a subset selection process is carried out in which the variables that truly provide information about the class are determined, i.e. the irrelevant variables that do not improve the accuracy of the model are removed. The subset selection method performed was *Wrapper* with forward selection [26]. Forward selection is an iterative method in which the model is initialized empty. In each iteration, the feature which best improves the model is included. This process is repeated

Table 1 Accuracy results of the wrapper algorithm used in the training phase.

	Accuracy of the set after including a new variable (%)				
Variable	$\{\emptyset\}$	$\{V_{14}\}$	$\{V_{14}, V_9\}$	$\{V_{14}, V_9, V_{16}\}$	$\{V_{14}, V_9, V_{16}, V_{15}\}$
$+V_1$	87.76	89.22	90.04	89.81	90.13
$+V_2$	88.72	89.71	90.28	90.14	90.51
$+V_3$	87.02	89.12	90.08	89.89	90.22
$+V_4$	88.78	89.78	90.35	90.19	90.52
$+V_5$	88.58	89.83	90.49	90.20	90.54
$+V_6$	88.67	89.69	90.20	90.07	90.40
$+V_7$	63.84	91.38	91.24	91.56	91.72
$+V_8$	71.14	91.56	90.80	91.48	91.53
$+V_9$	71.20	**91.62**	-	-	-
$+V_{10}$	88.93	90.10	90.57	90.45	90.74
$+V_{11}$	58.04	90.06	90.55	90.41	90.71
$+V_{12}$	64.54	91.43	91.25	91.59	91.71
$+V_{13}$	58.78	91.47	90.98	91.36	91.54
$+V_{14}$	**91.48**	-	-	-	-
$+V_{15}$	66.02	91.59	91.52	**91.94**	-
$+V_{16}$	69.24	91.06	**91.64**	-	-

until an addition of a new variable does not improve its performance. In our case, this performance (score) is measured with the NB classifier. Considering this method evaluates only a few subsets of variables, it is computationally efficient and robust against overfitting.

As can be seen in Table 1, the wrapper algorithm is performed until the fifth iteration, in which none of the remaining variables are able to improve the accuracy of the current set. The resulting dataset of each iteration is evaluated with the NB classifier with a 10-fold cross-validation. This classifier is based on the idea that an event occurs after other events that have an influence on it, but these events 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 $\{X_1, \ldots, X_N\}$, the following expression is satisfied:

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

As a result of evaluating the classifier in WEKA (82952 instances in total), an accuracy of 91.94% was obtained with the four features comprised by the model: cost in bits to encode the block (V_{14}), variance of the block of size 128×128 of the reconstructed image (V_9), lambda value (V_{16}) and the number of pixels of the frame (V_{15}). At first glance, it is possible to say that our model achieves a high accuracy. To verify this, the following section details the results obtained after analyzing the model with a large number of instances (test set) through a multitude of evaluation datasets in a real scenario.

5 Performance Evaluation

In this section the results of both the model and the implementation of the proposal in JEM are presented. For this purpose, the metrics necessary to perform the evaluation of the algorithm will be defined.

5.1 Evaluation of the Model

An exhaustive analysis has been carried out with the aim of evaluating the effectiveness of the generated model using the Random Access configuration. This analysis includes a test set for each sequence (from classes A1, A2, B, C, D, and E) and QP (22, 27, 32, and 37), making a total of 96 test sets, each of which contains an instance per complete block of size 128×128 pixels available in the sequence (note that the instances that were used to learn the model do not belong to these test sets). The decision obtained by the model for each block is compared with the original decision made by the JEM encoder for those same blocks. Consequently, it was necessary to decode each sequence in JEM to store the decisions made by the reference encoder at the 128×128 level. The said sequences, which are described in [23], are classified as follows:

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

Table 2 shows the accuracies reported by the model generated for each sequence and QP. As a result, it can be observed that a high accuracy is achieved in all cases, and slightly higher for QP 22, regardless of the class evaluated. In total, 5,987,832 instances were tested, with an average accuracy of 90.07%.

5.2 Simulation Process and Metrics

Our proposal has been evaluated in accordance with the conditions contained within the document of common conditions mentioned above [23], in which test conditions are set out to homogenize comparisons between experiments. Specifically, the QP values tested are 22, 27, 32 and 37, and the configurations are Random Access (RA), Low Delay B (LB) and Low Delay P (LP),

Table 2 Accuracy of the proposed model.

Class	Sequence	Accuracy (%)			
		QP 22	QP 27	QP 32	QP 37
A1	Tango2	87.28	86.62	86.45	86.15
	Drums100	96.05	88.09	89.21	88.98
	Campfire	95.34	90.40	91.97	91.52
	ToddlerFountain2	99.05	97.53	96.47	89.43
A2	CatRobot	87.03	88.47	89.98	91.29
	TrafficFlow	85.59	85.80	90.36	94.49
	DaylightRoad2	90.82	85.51	86.77	88.45
	Rollercoaster2	89.54	85.38	86.43	85.87
B	Kimono	91.83	90.11	89.66	89.73
	ParkScene	93.41	85.89	89.15	91.69
	Cactus	91.96	88.34	87.92	89.85
	BasketballDrive	92.76	87.99	90.17	89.62
	BQTerrace	97.43	84.40	85.47	93.00
C	BasketballDrill	96.98	91.23	88.01	83.79
	BQMall	93.87	91.13	90.46	88.57
	PartyScene	98.56	95.59	88.24	87.33
	RaceHorsesC	99.23	97.91	95.84	85.57
D	BasketballPass	98.02	93.32	87.18	75.12
	BQSquare	98.64	86.95	80.34	91.24
	BlowingBubbles	95.71	89.43	88.21	85.96
	RaceHorses	99.43	97.59	96.67	85.86
E	FourPeople	89.99	93.09	93.76	94.17
	Johnny	89.44	93.64	96.17	97.53
	KristenAndSara	89.66	93.05	95.35	96.34
	Class A1	94.46	90.68	91.04	89.03
	Class A2	88.26	86.29	88.38	90.00
	Class B	93.89	87.03	88.09	90.91
	Class C	96.76	93.39	90.07	86.52
	Class D	97.83	90.94	86.76	84.77
	Class E	89.70	93.26	95.09	96.01
	Average	91.81	88.65	89.76	90.04

all of them for the Main10 profile. The sequences mentioned in the previous subsection have been evaluated under these conditions.

The process performed to obtain the results is detailed below:

1. Encode the sequences with the HEVC reference software (HM 16.16 [17]) using the configuration files to compare JEM with HM provided in the JEM software (JEM 7.1 [5]), according to [23].
2. Decode each file with the HM decoder, generating both the raw video file of the decoded sequence and the statistical information for the model.
3. Encode each raw video file with the JEM encoder, where the coding process has been modified to integrate the proposal into the QTBT structure.

4. Compare the stream encoded with the reference JEM encoder with each proposed stream in order to obtain the coding efficiency and the time reduction of the proposal.

In order to compare the performance of the implemented proposal with the reference results obtained by the original encoder, the BD-rate and time reduction (TR) metrics have been used. The BD-rate metric evaluates the coding efficiency. A positive value means a penalty in bitrate of the proposal with respect to the reference encoder, maintaining the same image quality [8]. The TR metric measures the encoding time ratio of the two encoders, and is calculated as indicated in the following expression:

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

It should be noted that the time required to obtain the statistics used by the model is included in T_{proposal}. It represents only approximately 0.1% of the encoding time, so its impact is negligible in the resulting TR.

5.3 Experimental Results

Regarding the evaluation of the model developed, which was implemented on the JEM reference encoder (version 7.1) in order to compare the new coding flow based on the probabilistic algorithm of this proposal with respect to the coding based on the original QTBT structure of JEM. It should be noted that the fast large CTU (LCTU) technique [27] implemented in JEM has been disabled in the experiments, given that this tool skips the evaluation of certain blocks on the basis of the splitting decision of previously encoded CUs, and thus the evaluation would be distorted.

Table 3 shows the BD-rate and TR results for all the test sequences, which were evaluated under the common test conditions for the RA, LB and LP configurations. It can be observed that the sequences achieve good results in terms of BD-rate as the penalty obtained is lower than 1.35% on average, with a time reduction of more than 10% in all tested configurations. In addition, the results related to the coding time show that the implemented model performs better in high-resolution classes, that is, in classes A1, A2, B and E, as we can see in some cases the proposal achieves time savings of nearly 20%. For the low resolution of the test sequences belonging to the 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 quadtree are higher, it can be seen that the time reduction is lower compared with the rest of the classes, but with a negligible impact in terms of BD-rate penalty.

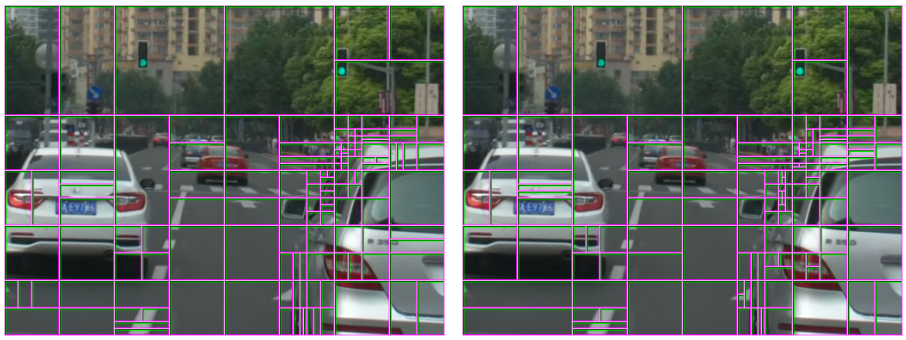
Regarding the performance of the proposed algorithm under different configurations, it can be seen that even though the model has been developed using statistical information obtained from the Random Access configuration, the results verify that the model obtained is generic, that is, the effectiveness

Table 3 Results of the proposal in RA, LB and LP configurations.

Class	Sequence	Random Access		Low Delay B		Low Delay P	
		BD-rate (%)	TR (%)	BD-rate (%)	TR (%)	BD-rate (%)	TR (%)
A1	Tango2	1.12	22.18	0.82	25.08	0.91	21.88
	Drums100	1.69	15.72	1.52	16.15	2.61	13.75
	Campfire	0.89	16.40	1.80	13.85	1.50	12.09
	ToddlerFountain2	0.42	8.27	0.33	6.65	0.44	5.93
A2	CatRobot	0.47	17.24	0.86	20.40	1.29	18.00
	TrafficFlow	1.03	16.89	0.37	21.70	1.91	16.52
	DaylightRoad2	1.00	19.47	0.76	20.05	1.69	18.24
	Rollercoaster2	1.02	23.39	0.39	25.14	0.57	21.55
B	Kimono	−0.50	16.19	0.26	15.26	0.65	11.00
	ParkScene	0.07	13.78	0.49	11.72	1.07	9.70
	Cactus	0.06	14.50	0.13	13.37	0.57	11.19
	BasketballDrive	1.06	14.65	0.75	11.82	1.35	9.87
	BQTerrace	0.29	11.58	0.84	12.10	1.24	9.93
C	BasketballDrill	0.63	8.99	0.44	6.92	0.73	5.23
	BQMall	0.29	9.09	0.20	6.96	0.46	5.46
	PartyScene	0.04	7.61	0.23	5.41	0.54	3.91
	RaceHorsesC	0.86	7.61	0.33	4.38	0.50	3.73
D	BasketballPass	2.09	4.24	1.48	2.60	1.75	1.88
	BQSquare	0.15	4.59	0.29	4.68	0.69	3.78
	BlowingBubbles	0.80	4.81	0.62	3.76	0.84	2.88
	RaceHorses	0.97	4.44	0.67	2.53	0.61	2.13
E	FourPeople	0.93	13.37	1.80	17.14	2.96	14.14
	Johnny	0.48	11.20	0.87	17.73	2.66	15.16
	KristenAndSara	0.86	13.80	1.39	19.73	4.85	16.66
	Class A1	1.03	15.64	0.87	15.43	1.37	13.41
	Class A2	0.88	19.25	0.60	21.82	1.37	18.58
	Class B	0.20	14.14	0.49	12.85	0.98	10.34
	Class C	0.46	8.36	0.30	5.92	0.56	4.58
	Class D	1.00	4.52	0.76	3.39	0.97	2.67
	Class E	0.76	12.79	1.35	18.20	3.49	15.32
	Average	0.70	12.50	0.74	12.71	1.35	10.61

of the model is similar for RA, LB and LP configurations. In fact, LB achieves, on average, the highest time reduction.

An illustrative comparison between the baseline transcoder and the one performed by the proposed algorithm is shown in Fig. 6. This figure displays the partitioning performed by the two alternatives in a portion of the 14^{th} frame of the DaylightRoad2 sequence for the RA configuration. As can be seen, the visual differences are minimal, since the partitioning achieved by our algorithm is nearly the same as the baseline reference, maintaining the visual quality of the image.



(a) Baseline. QP 27. (b) Proposed scheme. QP 27.

Fig. 6 Comparison of the CTU/CU partitioning between the baseline transcoder and the proposed approach.

5.4 Comparison with the Fast LCTU Encoding Tool

To the best of the authors' knowledge, there are no other HEVC-JEM transcoding proposals at the moment of writing of this manuscript. For this reason, this subsection compares the results of our proposal with the main fast encoding tool implemented in JEM, which, like the proposed model, omits the evaluation of the first partitioning level under certain circumstances. This tool, named fast LCTU [27], is a fast decision algorithm for CU depth used to speed up the encoding process in JEM when the maximum CTU size is set larger or equal to 128×128 pixels. When enabled, the encoder skips the R-D evaluation of certain CUs on the basis of the splitting decision of neighboring CUs. Therefore, it is a method that affects the splitting decision of the first partitioning level, and thus it was disabled in the evaluation of the proposal. This tool can be used, however, in an HEVC-JEM cascade transcoding scenario, which motivates its comparison with our proposal.

Table 4 shows the performance of the fast LCTU encoding technique. As can be seen, the achieved average TR is 5.74% for RA configurations, and slightly lower for LP and LB configurations. Regarding the coding efficiency, the BD-rate penalty is approximately 0.4% for all configurations. Compared with the proposed model, we can conclude that our proposal obtains more than two times the TR achieved by fast LCTU, and almost three times in the case of LB configurations, while also providing a good trade-off between coding efficiency and time savings. In addition, the proposed algorithm obtains higher TR values in both high-resolution A1 and A2 classes, which makes it more suitable for the next-generation video coding standard.

Table 4 Performance of Fast LCTU tool in RA, LB and LP configurations.

Class	Sequence	Random Access		Low Delay B		Low Delay P	
		BD-rate (%)	TR (%)	BD-rate (%)	TR (%)	BD-rate (%)	TR (%)
A1	Tango2	0.04	4.06	0.16	2.28	−0.01	2.78
	Drums100	0.20	5.12	0.19	3.63	0.20	3.94
	Campfire	0.47	9.54	0.65	8.18	0.96	8.91
	ToddlerFountain2	−0.03	6.14	0.16	5.08	0.20	5.59
A2	CatRobot	0.27	5.42	0.18	3.49	0.28	4.21
	TrafficFlow	0.77	4.11	0.31	2.87	0.31	3.00
	DaylightRoad2	0.51	5.33	0.50	3.08	0.46	4.45
	Rollercoaster2	0.04	4.85	0.08	3.55	0.14	3.47
B	Kimono	−0.39	4.42	0.20	3.37	0.04	3.50
	ParkScene	0.95	7.63	0.51	5.65	0.66	6.26
	Cactus	0.48	6.99	0.26	6.34	0.23	6.33
	BasketballDrive	0.23	7.73	0.32	5.95	0.23	6.43
	BQTerrace	1.60	7.44	1.36	7.03	0.48	8.54
C	BasketballDrill	0.31	6.38	0.10	5.25	0.01	5.49
	BQMall	0.25	5.39	0.23	3.88	0.16	2.66
	PartyScene	0.14	8.99	0.22	7.01	0.13	8.61
	RaceHorsesC	1.09	7.87	0.41	5.56	0.59	4.16
D	BasketballPass	0.26	4.23	−0.08	4.14	0.33	4.31
	BQSquare	0.42	4.82	0.29	4.48	−0.05	6.46
	BlowingBubbles	0.29	5.44	0.19	4.22	−0.16	5.54
	RaceHorses	0.29	5.04	0.62	3.41	0.15	4.56
E	FourPeople	0.51	4.70	0.23	4.93	0.35	3.72
	Johnny	0.46	3.07	0.54	3.40	1.19	3.56
	KristenAndSara	0.75	3.11	0.64	3.85	1.10	3.16
	Class A1	0.17	6.21	0.29	4.79	0.34	5.31
	Class A2	0.40	4.93	0.27	3.25	0.30	3.78
	Class B	0.57	6.84	0.53	5.67	0.33	6.21
	Class C	0.45	7.16	0.24	5.43	0.22	5.23
	Class D	0.32	4.88	0.26	4.06	0.07	5.22
	Class E	0.57	3.63	0.47	4.06	0.88	3.48
	Average	0.41	5.74	0.35	4.61	0.33	4.99

6 Conclusions and Future Work

In this paper, a CU partitioning decision based on an NB classifier for a video transcoder between HEVC and JEM is presented. The algorithm decides on splitting at the first level of the quadtree, that is, at the 128×128 pixel level.

By using the NB classifier, a decision model has been developed from features extracted from the coding and decoding of sequences in the HM reference encoder for the RA configuration. After applying a discretization and a subset selection process, we obtained a model composed of four variables, namely the variance of the block of size 128×128 pixels of the reconstructed image, the cost in bits to encode the block in the compressed HEVC stream, the number of pixels in the frame and the lambda value. This model obtains an accuracy

of 91.94% with 10-fold cross-validation, and an accuracy of 90.07% for all the test sequences.

Finally, the probabilistic model has been evaluated in a real scenario through a simulation process to compare the stream encoded with the reference JEM encoder and the codification flow of this proposal using the splitting decision at the first quadtree level. This comparison has been performed for RA, LB and LP configurations, and the results demonstrate a good trade-off between the complexity reduction and the encoding performance achieved by the proposal. The implementation of the algorithm obtains a TR of 12.50% with a BD-rate penalty of 0.70% for the RA configuration, for which it has been developed. However, the results for LB and LP show that the model obtained is generic, since the TR results are 12.71% and 10.61%, with penalties of 0.74% and 1.35% in the BD-rate, respectively.

As future work, new techniques and tools will be implemented to achieve higher time savings. While this proposal focuses on the first depth level of the QTBT, new approaches could be taken into account for acceleration of the remaining levels. On the one hand, the use of ML techniques has proven to be a good alternative to predict the splitting decision of the first level. On the other hand, the partitioning of the source HEVC bit stream could be used for the second level onward of the QTBT, given that the first level of the quadtree in HEVC matches the second level of the QTBT in JEM.

In addition to the partitioning structure, it would be also of interest to design fast and efficient techniques for other encoding modules, e.g. the intra prediction module, which could use the information of HEVC to predict the corresponding directional mode in JEM by using ML techniques.

References

1. CISCO (2016) Cisco Visual Networking Index - Forecast and Methodology (2015 to 2020)
2. ISO/IEC, ITU-T (2013) High Efficiency Video Coding (HEVC). ITU-T Recommendation H.265 and ISO/IEC 23008-2
3. ISO/IEC, ITU-T (2003) Advanced Video Coding for Generic Audiovisual Services. ITU-T Recommendation H.264 and ISO/IEC 14496-10
4. Ohm JR, Sullivan GJ, 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(12):1669–1684, DOI 10.1109/TCSVT.2012.2221192
5. JVET JEM Software Version - 7.1. "`https://jvet.hhi.fraunhofer.de/svn/svn_HMJEMSoftware/`"
6. Segall A, Baroncini V, Boyce J, Chen J, Suzuk T (2017) JVET-H1002 - Joint Call for Proposals on Video Compression with Capability beyond HEVC
7. Bross B, Chen J, Liu S (2019) Versatile Video Coding (Draft 5). Tech. Rep. JVET-N1001, Joint Video Experts Team (JVET)

8. Bjøntegaard G (2008) Improvements of the BD-PSNR Model. Tech. Rep. VCEG-AI11, ITU-T SG16 Q6
9. Vetro A, Christopoulos C, Sun H (2003) Video Transcoding Architectures and Techniques: an Overview. IEEE Signal Process Mag 20(2):18–29, DOI 10.1109/MSP.2003.1184336
10. Fernandez-Escribano G, Kalva H, Cuenca P, Orozco-Barbosa L, Garrido A (2008) A Fast MB Mode Decision Algorithm for MPEG-2 to H.264 P-Frame Transcoding. IEEE Trans Circuits Syst Video Technol 18(2):172–185, DOI 10.1109/TCSVT.2008.918115
11. Peixoto E, Izquierdo E (2012) A Complexity-Scalable Transcoder from H.264/AVC to the New HEVC Codec. In: IEEE International Conference on Image Processing (ICIP 2012), Orlando, FL, USA
12. Peixoto E, Macchiavello B, Hung M, Zaghetto A, Shanableh T, Izquierdo E (2013) An H.264/AVC to HEVC video transcoder based on mode mapping. In: IEEE International Conference on Image Processing (ICIP 2013),Melbourne, Australia
13. Peixoto E, Shanableh T, Izquierdo E (2014) H.264/AVC to HEVC Video Transcoder Based on Dynamic Thresholding and Content Modeling. IEEE Trans Circuits Syst Video Technol 24(1):99–112, DOI 10.1109/TCSVT.2013.2273651
14. Jiang W, Chen Y, Tian X (2014) Fast Transcoding from H.264 to HEVC based on Region Feature Analysis. Multimedia Tools and Applications 73(3):2179–2200
15. Díaz-Honrubia AJ, Martínez JL, Cuenca P, Gamez JA, Puerta JM (2016) Adaptive Fast Quadtree Level Decision Algorithm for H.264 to HEVC Video Transcoding. IEEE Trans Circuits Syst Video Technol 26(1):154–168, DOI 10.1109/TCSVT.2015.2473299
16. Franche J, Coulombe S (2018) Efficient h.264-to-hevc transcoding based on motion propagation and post-order traversal of coding tree units. IEEE Transactions on Circuits and Systems for Video Technology 28(12):3452–3466, DOI 10.1109/TCSVT.2017.2754491
17. JCT-VC HEVC Reference Software - Version 16.16. `https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/`
18. Chen J, Alshina E, Sullivan G, Ohm J, Boyce J (2017) Algorithm Description of Joint Exploration Test Model 7. Tech. Rep. JVET-G1001, Joint Video Experts Team (JVET)
19. An J, Chen YW, Zhang K, Huang H, Huang YW, Lei S (2015) Block Partitioning Structure for Next Generation Video Coding. Tech. Rep. COM16-C966, ITU-T SG16 Q6
20. An J, Huang H, Zhang K, Huang YW, Lei S (2016) Quadtree Plus Binary Tree Structure Integration with JEM Tools. Tech. Rep. JVET-B0023, Joint Video Experts Team (JVET)
21. Fayyad UM, Piatetsky-Shapiro G, Smyth P (1996) Advances in knowledge discovery and data mining. American Association for Artificial Intelligence, Menlo Park, CA, USA, chap From Data Mining to Knowledge Discovery: An Overview, pp 1–34

22. Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH (2009) The WEKA Data Mining Software: An Update. SIGKDD Explor Newsl 11(1):10–18, DOI 10.1145/1656274.1656278
23. Li X, Suehring K (2017) JVET-H1010 - JVET Common Test Conditions and Software Reference Configurations
24. ITU-T (2008) P.910 - Subjective Video Quality Assessment Methods for Multimedia Applications
25. Fayyad UM, Irani KB (1993) Multi-interval discretization of continuous-valued attributes for classification learning. In: Proceedings of the International Joint Conference on Uncertainty in AI
26. Guyon I, Elisseeff A (2003) An introduction to variable and feature selection. J Mach Learn Res 3:1157–1182
27. Chen J, Alshina E, Sullivan G, Ohm J, Boyce J (2016) Algorithm Description of Joint Exploration Test Model 2. Tech. Rep. JVET-B1001, Joint Video Experts Team (JVET)