



Machine Learning-Based approaches to reduce HEVC intra coding unit partition decision complexity

Maraoui Amna¹ · Werda Imen² · Bouaafia Soulef¹ · Sayadi Fatma Ezahra³

Received: 29 September 2020 / Revised: 31 August 2021 / Accepted: 13 October 2021 /
Published online: 5 November 2021

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2021

Abstract

The use of machine learning techniques to reduce recent video coding standards complexity such as High Efficiency Video Coding (HEVC) has received prominent attention. In fact, the fascinating HEVC standard coding efficiency gap is performed at the expense of dramatically increasing coding complexity. HEVC adopts a similar block-based hybrid video coding framework its predecessor H.264 Advanced Video Coding (H.264/AVC), but it provides a highly flexible hierarchy of unit representation, which includes three units: coding unit (CU), prediction unit (PU) and transform unit (TU). The recursive splitting of CU is content adaptive, which is one of the biggest differences compared to H.264/AVC. Adopting a large variety of coding unit (CU) sizes, the quadtree partition takes the lion's share of the HEVC encoding complexity, making it ever more challenging to use rigid traditional inference models to predict the efficient coding decisions. In this context, this paper investigates the resulting implications on both coding efficiency and encoding complexity, when using Fuzzy Support Vector Machine (FSVM) and convolutional Neural Network (CNN) models for partitioning in the HEVC intra-prediction. The first approach is an online FSVM-based algorithm designed to predict efficiently the CU partition module. The second one is a deep CNN method founded on a large-scale database of substantial CU partition training. Experimental results reveal that the proposed deep CNN approach, with 66.04% complexity reduction, outperforms the proposed online FSVM approach that achieves 45.33%. However, the FSVM with only 0.067% loss in coding efficiency compared to 1.69% engendered with the CNN, is considered as the approach that performs the best tradeoff between the compression efficiency and the complexity reduction when optimizing the HEVC complexity at all Intra configuration.

Keywords HEVC · Deep CNN · Online FSVM · Complexity reduction

✉ Maraoui Amna
maraouiamna@gmail.com

¹ Electronics and Microelectronics Laboratory, Faculty of Sciences of Monastir, Environment Street, Monastir 5019, Tunisia

² Electronic and Information Technology Laboratory, University of Sfax, Sfax, Tunisia

³ Networked objects, control and communication systems Laboratory, University of Sousse, Sousse, Tunisia

1 Introduction

The huge amount of digital video data, which require high compression rates for electronic devices to manipulate it, transmit it in real time, and store it as needed has become even more latent in recent years. By focusing on this problem, industry and academia have conducted extensive research in recent decades on more advanced and efficient video compression techniques. H.265 / HEVC [25], defined by the Video Technology Collaboration Team and launched in 2013, has recently become an advanced video compression and is gradually replacing its predecessor, the H.264/AVC standard [28], in commercial applications and multimedia devices. Compared to H.264/AVC, HEVC provides higher compression ratios of 40% to 50% but up to 500% increase in complexity [6]. Since it is a hot research topic, several approaches are conducted to reduce the HEVC encoder complexity as illustrated in Figure 1.

Most HEVC optimization researches are interested in inter mode optimization [4, 9, 10, 23, 30, 35]. Usually, the fast-encoding algorithms suggested to optimize HEVC all intra configuration focus, especially on optimizing the intra-prediction mode selection [8, 11] and the CU partition process [2, 14, 16, 17, 19–21, 27, 31–34].

Indeed, one of the most important and complex module introduced in the HEVC standard is flexible quadratic tree based CU partitioning structures. Initially, each video frame is divided into a series of equally sized square blocks called Coding Tree Units (CTUs), which are used as roots of quadratic partitioning trees called Coding Trees. As displayed in Figure 2, the coding trees pursue a recursive partitioning scheme, in which each node, called the coding unit, can be subdivided recursively into four new nodes until it reaches the smallest possible size of CU. In general, HEVC allows 8x8, 16x16,

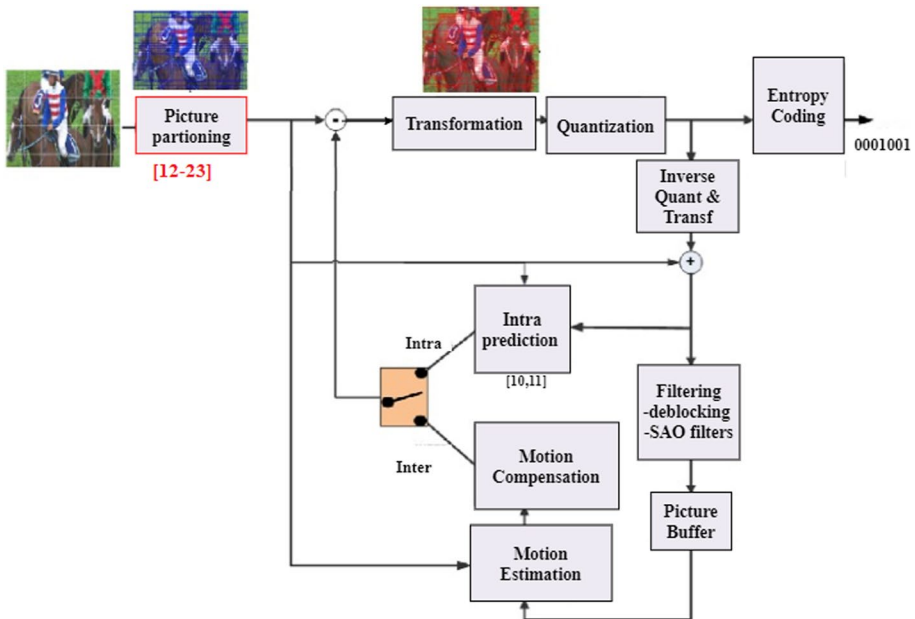


Fig. 1 Block of interest within HEVC block diagram

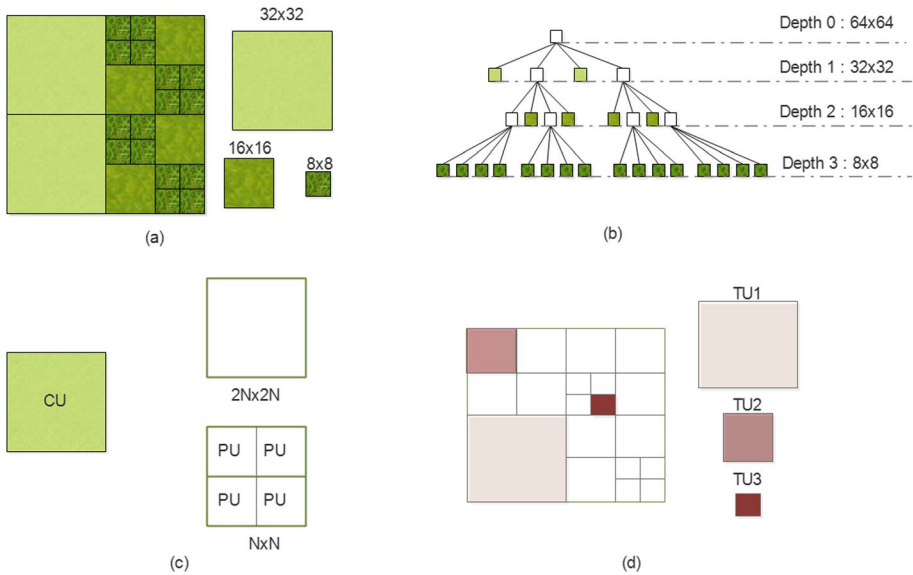


Fig. 2 (a)CTU Structure partition, (b) PU sizes for CU intra prediction. (c) hierarchical depth of a CTU divided into various CU sizes. (d) TU structure

32x32 and 64x64 CUs. Once the partitioning structure of the CU has been completed, each CU can then be split, at the prediction stage, into one or more prediction units with several modes. In addition, at the transformation stage, each CU can be split into one or more recursive transformation units. The recursive Rate-Distortion (RD) cost comparisons strategy consists in testing all possibilities up to find the best. Both the sizes of PU and TU cannot exceed the size of CU. Due to recursive division, the encoder has to scan all combinations of all possible sizes of CU, PU, and TU to select the optimal solution, which is very time consuming. In addition, an intra 4x4 TU has to decide whether to skip transform or not [15]. This extremely complex strategy is the basis for determining the optimal CU, PU, and TU. If we consider that prediction and transforms are computed several times in each CTU, it becomes evident that the mode decision selection based on RD cost is one of the main culprits for HEVC complexity.

Therefore, several studies found in the literature focused on addressing this problem by designing fast mode decision algorithms. Most of them are based on heuristics resulting from statistical video characteristics analyzes. These approaches eliminate costly steps yielding to significant complexity reduction at the expense of coding efficiency performance degradation [17, 21]. The CU structure within HEVC can support several sizes, which significantly reduces the differences between the extracted CUs and penalizes the compression efficiency performance of algorithms based on statistical correlations. Recent research on video coding has shown that Machine Learning-based models such as Support Vector Machine [19, 32–34] and Neural Network [2, 14, 16, 20, 27, 31] are deeply used to speed up the mode decision process while maintaining high coding performance.

The tradeoff combining complexity reduction and compression efficiency is relevant to fairly compare performances of several approaches. In the literature, results achieved using

learning-based methods outperform those of statistical-based approaches. This observation guided us to continue our research in the direction of FSVM and CNN approaches in order to reduce the HEVC complexity at all intra configuration.

In this paper, we focus on designing a fast coding unit decision algorithm reducing HEVC CU partition structure complexity. Thus, we exploited two major machine-learning techniques to contribute to all HEVC intra configurations complexity reduction, while maintaining the compression efficiency and video quality performance. We developed an innovative FSVM approach and three well-trained CNN structures to predict HEVC coding unit partition size for all intra configurations. We also discussed the performance results in this paper.

The remainder of the paper is organized as follows. Section 2 introduces a review of related works. Section 3 displays motivations based on statistical analysis. Sections 4 and 5 describe the proposed FSVM and CNN approaches to reduce the HEVC all intra configuration complexity. We discuss the experimental results in section 6. Finally, section 7 concludes this paper.

2 Related works

Existing works that focus on the fast CU sizes selection can be broadly classified into two categories: methods based on correlation analysis and classifications founded on machine learning approach. CU fast decision algorithms designed based on statistical information aim to closely extract block homogeneity indicators in order to predict faithfully CU size decisions. In [17, 21] gradient information of a CU block is used to derive its texture complexity. Kernel based gradient calculation is a time consuming operation, which explains unexciting optimization results. Hence, in [21], Mallikarachi et al. proposed a low complex texture analysis technique based on local range values and variance of a given block. This approach provides the required information on texture homogeneity to decide on CU splitting at an early stage and allows an average of 58.58% reduction in HEVC encoding time. However, this algorithm renders less content-adaptive by the fixed threshold and rigid decision trees, leading to inefficient CU size decisions for arbitrary sequences translated into a penalizing increase in the compression performance degradation. In fact, the block size partitioning flexibility within HEVC considerably reduces the difference between several coding units, which explains the deficiency of fast algorithms based on statically analyzed correlations to perform considerable complexity reduction results with the degradation of the compression efficiency performance.

Recently, with the popularization of the Artificial Intelligence area and the computational power to process large amounts of data, several approaches have emerged aiming to solve the problems derived from video coding through the support of Machine Learning and CNN techniques.

In fact, to provide optimal discriminatory solutions to the problem of CU size selection, approaches based on machine learning used supervised learning algorithms to learn the model parameters using data obtained from encoded video sequences. In this issue, Jun et al. [32] suggested a fast block-partitioning algorithm using a Support Vector Machine (SVM) classifier. In each CU depth and depending on four extracted features, three offline trained SVM CU splitting models are loaded to predict the class label of the current CU.

Experimental results showed that the proposed algorithm reduced the computational complexity by up to 53.9% when disabling the Rate Distortion Optimization (RDO) process and lead to 1.27% loss in coding efficiency. Yun et al. [34] proposed a machine learning-based fast coding unit depth decision method, which optimized the complexity allocation at the CU level with a given rate-distortion cost constraints. They modeled the quad-tree CU depth decision process as a three-level hierarchical binary decision problem. Then, they derived a sophisticated RD-complexity model to determine the optimal parameters for a three-output joint classifier, which was capable of minimizing the complexity in each CU depth at given RD degradation constraints. Experimental results indicated that the proposed fast intra coding algorithm attained about 51.45% encoding time reduction on average with only 1.98% Bit-rate (BD) increase. For the binary classification problems, SVM classification is often regarded as a common machine-learning algorithm that can also mediate the resulting additional complexity of the encoding loop. In this regard, [33] applied two linear SVM models that use the difference in depth and the cost ratio of Hadamard transform (HAD) and RD as features to perform early CU split and early CU termination decisions. Experimental results revealed that the proposed fast intra coding algorithm achieved about 54% encoding time reduction on average with only 0.7% BD-rate increase. Focusing on complexity classification using machine-learning technology, the authors in [19] proposed an adaptive fast CU size decision algorithm for HEVC Intra prediction. First, to describe the CU complexity, features are extracted. Then, SVM is performed to evaluate and construct the CU classification mode complexity. Finally, depending on the classification complexity, a suitable adaptive fast CU size decision algorithm is released. The experimental results showed that the proposed algorithm could achieve around 60% encoding time reduction on average with only 1.26% increase in the coding efficiency. All these algorithms achieve interesting encoding time improvements, performed at the expense of the rate- distortion performance degradation.

With an automatically feature extraction process, the convolutional neural network proved to be an effective method in extracting the codec behavior. Kuanar et al. [14] suggested a CNN based algorithm, which wisely extracts image features and performs a classification job. These classifications' results are later used in the encoder downstream systems to find the optimal CUs in each of the block trees and reduce the CU size prediction complexity. The experimental results indicated that the proposed algorithm reduces the encoder time by saving up to 66.89% with a minimal coding efficiency loss of 1.31%. Li et al. [16, 31] proposed a deep CNN approach to learn to predict the optimal CU partition instead of conventional brute-force RDO search. Experimental results revealed that the approach reduces intramode encoding time by 62.25% with a coding efficiency increase of 2.12%. Chenying et al. [27] proposed a two-stage approach. The rough early-termination algorithm, which is based on the range of CU luma samples and directly related to CU partition mode, represents the first stage of the suggested approach. To accurately estimate the partition mode of ambiguous CUs conveyed by the first step, the authors designed and trained CNNs in the second step. In [20], CNN was recommended to assist in the early choice of CU size in HEVC intra-coding. The algorithm proposed by Liu et al. does not rely on the spatial information, making it easier to parallel the execution of the algorithm. Experiments showed that an average intra-encoding time of 61.1% was saved, while the compression efficiency increased by 2.67%. CNN-based algorithms improved the encoding time more than SVM solutions. However, the rate-distortion performance degradation is more penalizing.

3 Motivations

In HEVC intra coding, the main cause of the high complexity is that a frame is treated as different square blocks. The largest block size in HEVC is 64x64 while the smallest block size is 4x4. Unlike the 16x16 largest block in H.264/AVC, this promoted flexibility allows HEVC to have a better adaptation to different video content. This is the main point that reduces the compression efficiency performances of statistical correlations based algorithms compared to machine-learning approaches. These observations motivate us to focus our optimization on the FSVM and CNN approaches to reduce the HEVC complexity to all intra configurations.

We carried out explorations to estimate the potential computational cost that the encoder spent on exploring the best CU decision. We stored all information about the CU partition in a Best CU Decision Trace (BCDT) file when encoding the video sequences with the HEVC test Model (HM) encoder version HM16.5 [12]. Then, in the second step, the encoder will read the best CU decision directly from the BCDT file without any trials, instead of exploring the best CU partition by exhaustive search. We performed experiments on four sequences with different characteristics including Cactus (1920x1080), BQMall (832x480), BlowingBubbles (416x240), and KristenandSara (1280x720). For each sequence, the time reduction compared to the original, noted ΔT , was determined as exposed in Table 1.

$$\Delta T = \frac{1}{4} \sum_{i=1}^4 \frac{T_{prop}(QP_i) - T_{org}(QP_i)}{T_{prop}(QP_i)} \times 100 \quad (1)$$

Where T_{org} indicates the encoding time of the original platform, T_{prop} represents the encoding time of the proposed algorithm and QP_i refers to four applied QPs (22,27,32,37). It should be noticed that if classifiers predict CUs size correctly, the computational complexity reduction can reach 60% on average without modifying the video quality and the compression efficiency performances. This is the upper bound of computational complexity reduction can reach with the same compression efficiency and video quality performances. This analysis demonstrates that an interesting optimization potential is obtainable when providing useful insights for fast intra CU decision algorithm design.

4 Proposed Algorithm based on Fuzzy Support Vector Machine (FSVM)

As previously mentioned, HEVC supports four Intra CU sizes ranging from 64x64 to 8x8. Each CU could be split recursively into four equally sized sub-CUs until it reaches the tiniest CU size. The best CU partition is extracted by comparing the RD cost between the CU

Table 1 Reduction of encoding time when CU decisions are given

Resolutions	Sequences	ΔT (%)
1920x1080	Cactus	66.6
832x480	BQMall	59.2
416x240	BlowingBubbles	43.9
1280x720	KristenAndSara	73.8
Average		60.8

and its four sub-CUs in each depth level. For this instance, the computational complexity seriously restrains real time applications performances.

The intra coding process goal is to improve coding efficiency when exploiting spatial redundancy within the same frame. This explains the fact that homogeneous regions in a frame are more expected to be coded by large CU block size. However, small CU block size is obviously selected to code regions with detailed texture. Founded on this observation, if we worthily evaluate homogeneity complexity of a given video frame, the intra prediction process can be deeply optimized. Indeed, based on frame complexity evaluation, some of the original CU candidates could be skipped to select small CU, or the original CUs splitting could be terminated early with the selection of a large CU, which could significantly reduce HEVC encoding speed performance.

4.1 Background on Fuzzy Support Vector Machine

SVM is a powerful machine learning tool, since it can perform binary classification problems with significant computational advantages. Even if SVMs are widely used in the CU split decision, modeled as a binary classification problem, they are still sensitive to outliers considered as noise.

Undoubtedly, using SVM, gives more confidence in the classification of a test sample when it is far away from the hyperplane: if the test sample does not belong to a fixed risk area, as used in [22], the output will be accepted, otherwise there is a high risk of misclassification and the uncertain output will be rejected. To prevent such a problem, a CU decision method based on FSVM [18] is adopted in this paper. Compared to SVM, FSVM adopts additional regulation parameters noted S_i to reduce outliers which are eliminated in training samples noted as C_1 and C_2 in Figure 3. The final FSVM provided set of learning samples is labeled as follows:

$$(y_1, X_1, S_1), \dots, (y_{n-1}, X_{n-1}, S_{n-1}), (y_n, X_n, S_n) \tag{2}$$

For $i=1, \dots, n$, in addition to a decision label $y_i \in \{-1, 1\}$ associated to each training sample $X_i \in R^N$ proposed in SVM, a fuzzy membership S_i representing the attitude of the corresponding point X_i toward one class is proposed with FSVM where $0 < \sigma \leq S_i \leq 1$, while σ is as small as possible. Using FSVM, the optimal hyperplane problem is then regarded as the solution to

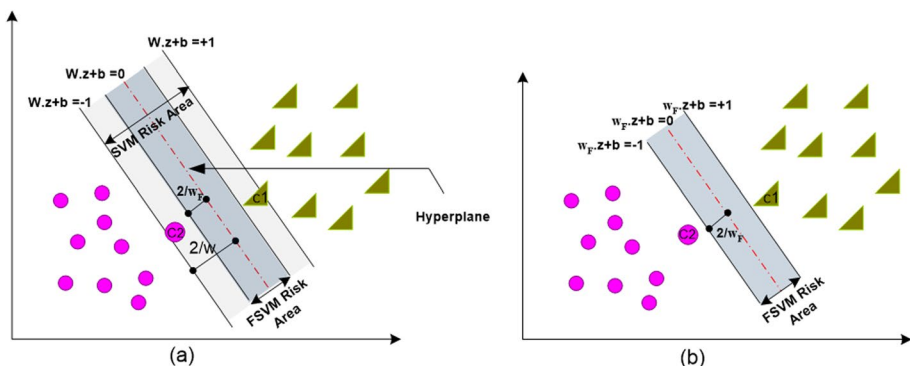


Fig. 3 (a) risk area SVM (b) risk area FSVM

$$\begin{aligned}
 & \text{Minimise} \quad \frac{1}{2}w * w + c \sum_{i=1}^n S_i \xi_i \\
 & \text{Subjected to} \quad \begin{cases} y_i(w \times Z_i + b) \geq 1 - \xi_i, & i = 1, \dots, n \\ \xi_i \geq 0, & i = 1, \dots, n \end{cases} \tag{3}
 \end{aligned}$$

Where c is the regulation parameter which controls balance between margin maximum and false classification prediction. $Z_i = \Phi(x)$ denotes the corresponding feature space vector with a mapping from R^N to a feature space Z , ξ_i a measure of the error for the FSVM. S_i ξ_i is a measure of the error with different weighting and (W, b) represent the hyperplane where W is the weight vector and b is the intercept of the hyperplane. The decision function is described as:

$$y(x) = \begin{cases} +1 & (W \times Z + b) > 0 \\ -1 & (W \times Z + b) \leq 0 \end{cases} \tag{4}$$

To eliminate the output of test samples that belong to the risk area, the decision function is redesigned as [22]:

$$\begin{cases} 0 & -0.005 \leq \frac{wZ+b}{\|w\|} \leq +0.005 \\ y(x) & otherwise \end{cases} \tag{5}$$

The weight assigned to the samples used to process FSVM classifier training, depends on their individual contributions. The weight S_i of the i th sample, considered as outlier, can be used as for slack variable. Based on distance between the current X_i and central \bar{X}_i samples. S_i is calculated as follows:

$$S_i = \frac{2}{1 + \exp(0.1\|X_i - \bar{X}_i\|)} \tag{6}$$

4.2 Image features selection

The purpose of this algorithm is to model the quadtree structure of the CTU partitions as a cascade binary classification task. Three classifiers $C_{64 \times 64}$, $C_{32 \times 32}$ and $C_{16 \times 16}$ are set to decide whether to split (S) or not to split (N-S) the considered CU as described in Figure 4.

The feature extraction is crucial since it may directly decide the bound of the proposed algorithm performance. Thus, to achieve a better accuracy performance, it is vital to delicately select features that affect CU size decisions. With an experimentally selected optimal feature set planned in Table 2, the accuracy performance can be further mended. The same features are used for 3 classifiers.

- Variance, x_var , is a significant feature, which can efficiently evaluate the video frame texture and complexity. Based on the current CU variance feature value, CU complexity is evaluated to learn the CU quad-tree partitioning.

$$x_var = \frac{1}{N^2} \sum_{i,j=0}^{N-1} [P(i,j) - \bar{P}(i,j)]^2 \tag{7}$$

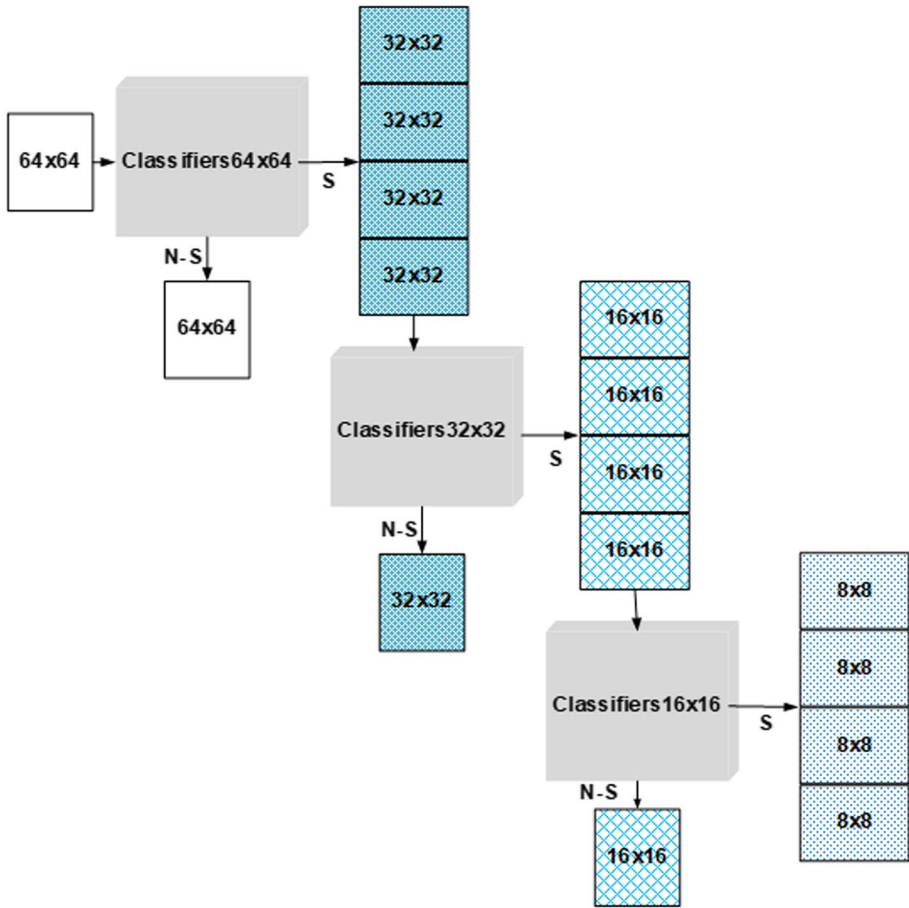


Fig. 4 Cascade classifiers

Where N represents the current CU size, and $P(i,j)$ and $\bar{P}(i,j)$ represent the luminance and the mean luminance values of the current CU, respectively.

Table 2 Description of the selected features

ID	Selected Feature	Feature Explanation
1	X_var	Variance of the image pixel
2	x_depth	Mean depth of the adjacent CUs
3	x_partition	Mean of neighbors PU mode (intra_NxN,intra_2Nx2N)
4	x_CBF	Mean of neighbors Coded Block flag
5	x_RDCost	RDCost
6	x_bits	Coding bits
7	x_Distortion	Distortion
8	x_QP	Quantization Parameter(QP)
9	x_CtxSkipFlag	Intra transform skipping Flag of left and above blocks

Nearby CUs typically retain identical textures in natural images. Obviously, the current CU’s optimum depth level can have a strong correlation with its neighboring CUs. The selected spatial-temporal correlations features are: We use the mean depth of the adjacent CUs as illustrated in Figure 5, which are upper, upper-right, upper-left, left and co-located CUs depth, as a feature x_depth to predict the current CU quad-tree partitioning decision.

$$x_depth = \frac{1}{5} \sum_{i=0}^5 x_depth(i) \tag{8}$$

The partition size information $x_partition$ feature, informs about the selected neighboring PU size from intra mode candidates. $x_partition$ is the mean value of upper, upper-right, upper-left, left and co-located Cus $x_partition(i)$.

$$x_partition = \frac{1}{5} \sum_{i=0}^5 x_partition(i) \tag{9}$$

The coding information flag x_CBF feature provides the mean of decision on TU partition selected for neighboring upper, upper-right, upper-left, left and co-located Cus $x_CBF(i)$.

$$x_CBF = \frac{1}{5} \sum_{i=0}^5 x_CBF(i) \tag{10}$$

Features that deeply affect RD coast performance are x_RDcost, x_Bits and, $x_Distortion$. Obviously, x_QP feature corresponding to Quantization Partition (QP) it is chosen to adjust between the coding bits and distortion, when selecting the CTU partitions. Finally, since the intra transform skip in HEVC can be applied to TUs of 4x4 size only [15]. $x_CtxSkipFlag$ is a flag used to detect if an intra transform skip is applied on the neighboring upper and left blocks.

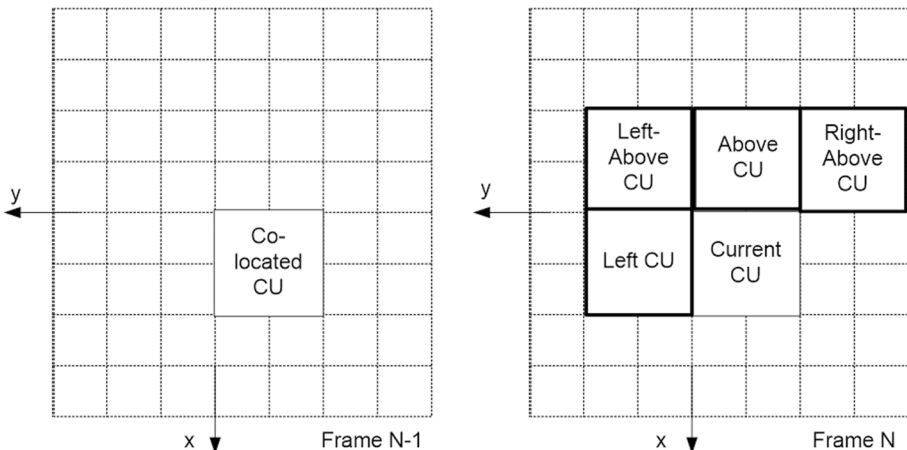


Fig. 5 Current 64x64 CU and its Neighbors used to calculate feature’s mean

To select the best performing features, we proceeded in three steps: in the first step, we were inspired by the recent related works [19, 32] on fast video coding. We summarized them and came up with the intersection of feature candidates for the CU depth prediction, which are the variances x_var and x_QP . To evaluate several features combinations impact on the coding efficiency, and the complexity reduction's performance, we respectively used the Bjøntegaard delta bit-rate (BD-BR)[3] and ΔT in eq 1.

As illustrated in Table 3, the use of these two features allows a reduction of 37% in complexity against a degradation in compression efficiency that reaches 6%. Furthermore, the related works [19, 32] used different operators to generate features representing the direction complexity and the edge detection within a frame. Indeed, in [19] the authors adopted the operator of Sobel to develop several diagonal directional Sobel Operator components to estimate the direction complexity of the Intra video frame. This approach allows to extract a reliable representation of the frame complexity direction symbolized by the feature (DCom) used in [13]. Other authors found that the Sobel and Roberts operators are simple to calculate, but their detection performance is not so reliable [32]. They also discarded the use of the Canny operator due to its complexity and opted to use the dominant edge detection method (DEA) proposed by [26] as a feature to detect edges.

To overcome these additional calculations, we used variables already calculated by the standard, (x_depth), ($x_partition$) and (x_CBF and $x_CtxSkipFlag$), which respectively gave an idea about the CU, PU and TU selected sizes of the neighboring blocks. The integration of these features in a second step revealed the increase of the optimization rate, which reached 47.1% against an obvious degradation of the compression performance. We therefore opted for the integration of three additional features $x_Distortion$, x_RDCost , and x_bits , which influenced the compression performance. The results are shown Table 3. The combination of the nine features reduces the encoder complexity by up to 41% with a negligible degradation of the compression efficiency.

The performance indicators for our FSVM system are mainly: accuracy, precision and sensitivity. In our application scenario, if the predicted output or the ground-truth mark is the same as the targeted decision, it is denoted by +1, otherwise by -1. We considered four potential outcomes when passing a sample into a model in the testing process. If the prediction is +1 and the mark is also +1, the True Positive outcome is called (TP). If the prediction is +1 while the mark is -1, the False Positive result is (FP). Similarly, if the prediction is -1, two other outcomes are called False Negative (FN) and True Negative (TN), based on the different mark values. We can calculate the accuracy performance, as expressed in eq.11, to measure the general performance of a classification model based on the four results we defined above. The precision of making a targeted decision is defined in eq.12. The number of correct positive decisions (TP) divided by the number of all

Table 3 Features extraction steps results

Sequences	Step 1		Step 2		Step 3	
	$\Delta T(\%)$	BD-BR(%)	$\Delta T(\%)$	BD-BR(%)	$\Delta T(\%)$	BD-BR(%)
PeopleOnStreet	-29.39	4.8	-36.8	6.3	-29.21	-0.072
BasketBallDrill	-41.85	3.2	-24.3	-0.04	-39.86	-0.012
BQSquare	-35.38	0.5	-20.4	0.69	-44.44	0.016
FourPeople	-37.33	6.0	-47.1	6.38	-41.08	0.028

samples predicted to be positive (TP+FP). The model's sensitivity computes the number of actual positive samples that can be successfully captured by a classification model as defined in eq. 13.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (11)$$

$$Precision = \frac{TP}{TP + FP} \quad (12)$$

$$Recall = \frac{TP}{TP + FN} \quad (13)$$

The overall accuracy of the classification prediction is 85%, where precision and recall are respectively 84% and 88%.

4.3 Overall Algorithm

The main idea of the proposed algorithm is to online extract the ground truths and the selected features for the model training from the first ten frames of the input sequences, while being encoded by the original HEVC test model. As described in Figure 6, FSVM resumed in eq. 3 is then used to train and periodically Update the classifiers C_64x64, C_32x32 and C_16x16 for successive frames. The CU split decisions of the following intra frames are extracted, once the classification task is finished. The original HEVC test model will be launched to extract the CU of risk zone samples. Otherwise, classifiers outputs will be directly used for the encoding process. This feedback step is crucial to reduce the computational complexity of the HEVC encoder without affecting its coding performance. The choice of an online training process improves the prediction accuracy since training and testing sequences are successive.

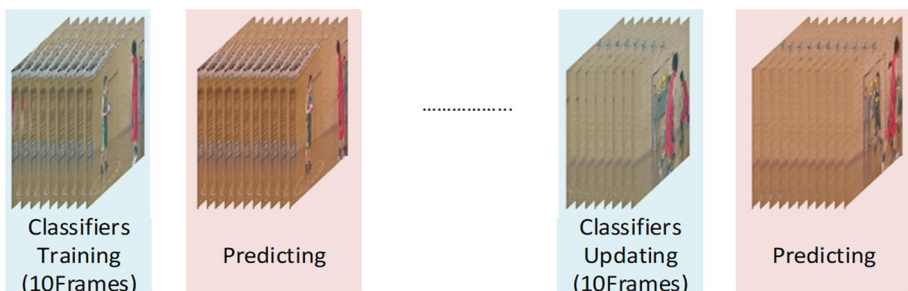


Fig. 6 classifiers training and updating for the FSVM proposed method

Table 4 Example of a lengthy table which is set to full textwidth

Depth level	Number of Samples	
	Splitting	Non-Splitting
0	9323733	3086267
1	18323303	18508377
2	34605454	38718334
Total	62252490	60312978

5 Proposed Fast CU decisions based on CNN

While studying the previous works related to deep learning solutions, we concluded that there is room for improvements in CNN-Based solutions. Complexity reduction performance from [16] is promising, but it is penalized by a significant compression efficiency degradation. Based on a more sophisticated database, a new CNN-based approach is proposed to enhance all the intra HEVC encoder performance tradeoff between the compression efficiency and the complexity reduction. Indeed, a fine modification of CNN data training models proposed in [16], is likely to produce better long run results.

5.1 Database establishment

In this step, the database for the Intra-mode HEVC CU Partition is established from images with different resolution (4928x3264, 2880x1920, 1536x1024, 1792x1024 and 768x512) at different QPs (22,27,32,37) and encoded using the HEVC reference software HM[12] with the configuration file encoder_intra_main.cfg. A total of 2500 images are randomly divided into training (2125 images), validation (125 images) and test (250 images) ensuring sufficient and diverse training data for learning to predict CU Partition.

The extracted database is divided into three folders: training, validation and test, containing all CUs sizes with their corresponding binary labels for different QPs. A total of 122,565,468 labeled samples are obtained, as listed in Table 4 The number of split Cus

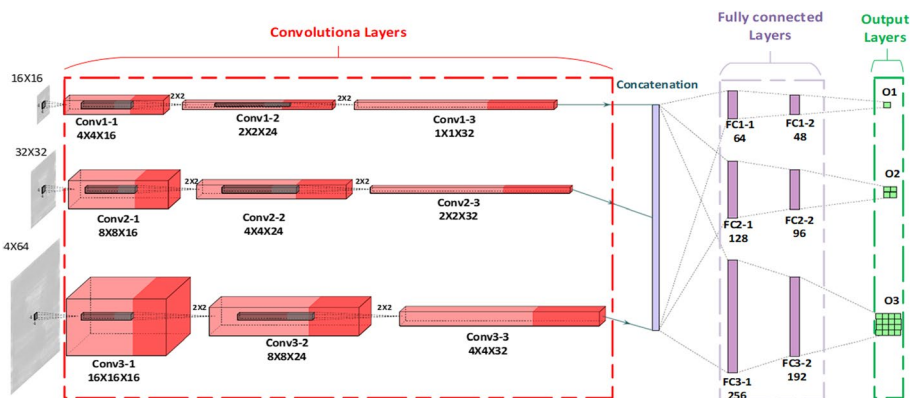


Fig. 7 CNN architecture

samples is close to that of non-split one over the whole database, which ensures the sufficiency of our database.

5.2 CNN architecture

As shown in Figure 7, three separate CNN models are learnt to provide classifiers at three levels sharing a uniform CNN structure with different kernel sizes. Each CNN model is composed of an input layer, three convolutional layers, a concatenation layer and three fully connected layers.

The corresponding luminance block component is loaded as an array of pixel values. Then, it is convoluted with 4×4 kernels (16 filters in total), with stride 4 at the first convolutional layer. Learned low-level features for the CU partition extracted from this layer represent information on edges detections and orientations in the considered input data. During the second and the third layers, data are sequentially convoluted twice with 2 x 2 kernels, with stride 2, (24 filters for the second layer and 32 filters for the third layer) to generate features at a higher level. Global and local features are collected from the second and third convolutional layers. Next, the vectorized features of the concatenating layer are processed in three models, to pass through three fully connected layers: two hidden fully connected layers successively generate features vectors, and one output layer extracting O_1,O_2 and O_3 outputs containing respectively 1, 4 and 16 binary elements. After the first fully connected layer, extracted features during the training phase are randomly dropped out [24] up to 0.5. This probability is reduced to 0.2 for the second fully connected layer.

The overall algorithm process is described in Figure 8. In a first step, extracted ground-truth is used as inputs of the training step. Then, three CNN models are fed with an entire CU, with three different sizes for each model, and each CU is able to hierarchically

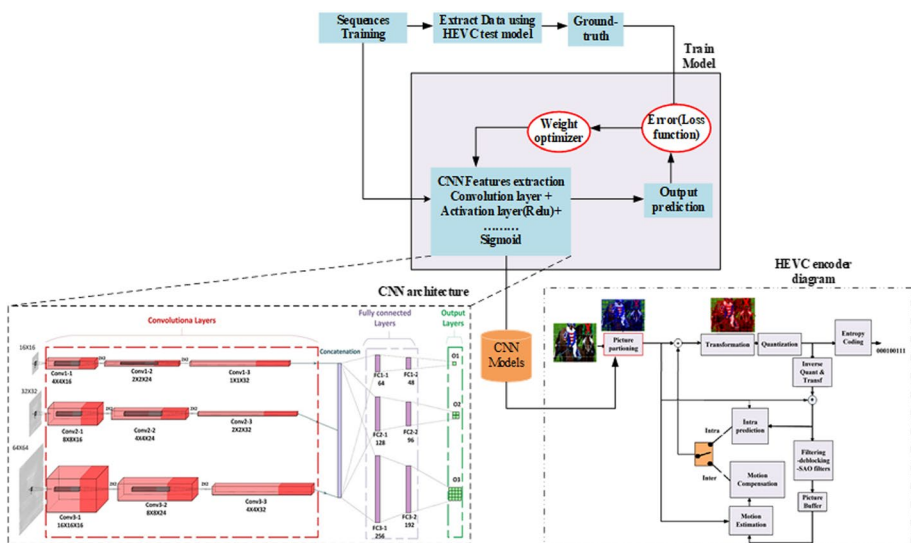


Fig. 8 Global structure of CNN based coding unit prediction approach: Database extraction, Training model, CNN architecture and HEVC encoder diagram

Table 5 Parameter setting for CNN training

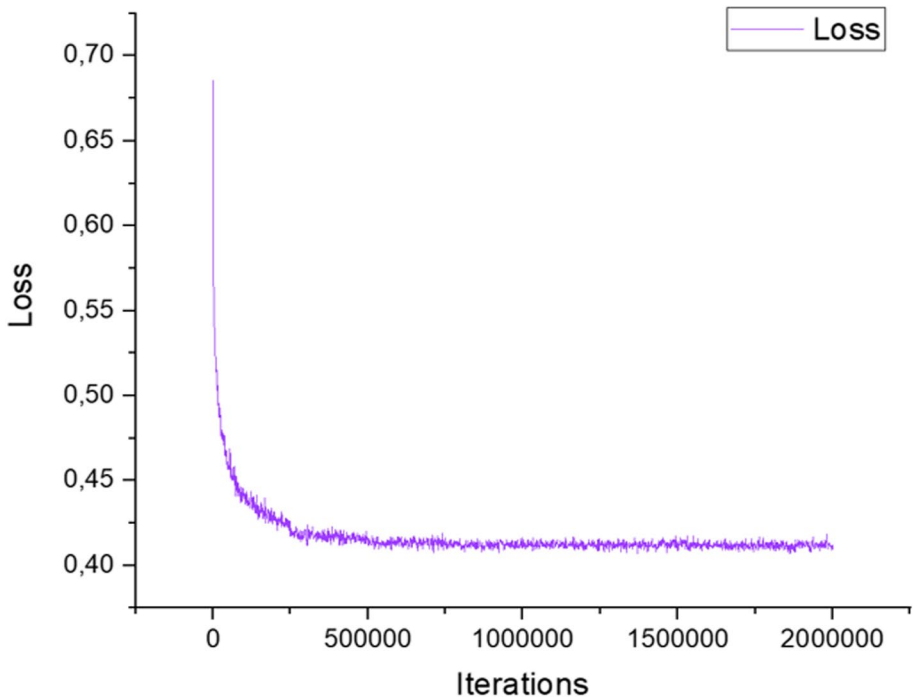
Input size	64x64
Label output(size)	64x64
Training Samples	122,565,468
Learning Rate	10^{-2}
Weight Decay	0.3163
Momentum	0.9
Batch size	64
Number of Iterations	2,000,000

generate a branch of structured output, representing all predictions. After the training process, a three CNN models are generated and integrated in the HEVC encoder to predict the CU partition.

Given the described CNN architecture, we use the well known Tensorflow framework [1] to learn the CNNs. The parameters for the learning process are summarized in Table 5.

To optimize our CNN weights, we used the back propagation algorithm, which is based on a stochastic gradient descent solver with 0.9 momentums to train the CNN models. The initial learning rate was 0.01 and decreased by 1% exponentially every 2000 iterations and there were in total 2000000 iterations.

Then, the CNN can be trained by optimizing the loss function, which is calculated as eq. 14, overall the training samples.

**Fig. 9** Training loss

$$L_l = \frac{1}{k} \sum_{m=1}^k y_l^{(m)} \log O_l^{(m)} + (1 - y_l^{(m)}) \log (1 - p_l^{(m)}) \quad (14)$$

Where y_l denotes the truth-ground classification labels, O_l designates the predicted output and K indicates the batch size to train CNNs. The obtained result after training is illustrated in Figure 9. We can see from this figure that the loss converges after 5×10^5 iterations.

6 Experimental results

6.1 Experimental setup

To evaluate the performance of the proposed algorithms, several experiments are conducted on the HEVC test model version HM16.5 [12]. Since only the complexity reduction of the HEVC intra-coding process is considered, we made use of the configuration file encoder_intra_main.cfg. We carried out our experiments on 17 test video sequences, listed in Table 6, from the Joint Collaborative Team on Video Coding (JCT-VC) standard test set [29]. To evaluate the coding efficiency, the complexity reduction performances and the subjective video quality, we respectively used the BD-BR, ΔT detailed in eq. 1 and the Bjøntegaard delta peak signal to noise ratio (BD-PSNR).

All experiments are performed on Intel(R) Core(TM) I7-8700 CPU@3.20 GHz, 16 GB RAM computer with Windows 10 professional 64-bit operating system. For FSVM implementation, the optimized libSVM [5] library was used with the Gaussian Radial Basis Function (RBF) kernel. Several functions from tensorflow [1] library are used in our CNNs implementation and the speed Geforce GTX 1050 Ti GPU was activated to accelerate the training. CNN models are trained on 3 classifiers at 4 QPs values (= 22, 27, 32, 37).

Table 6 Sequences and its related information grouped by class

Class	Resolution	Sequences	Frame Rate	Number Frames
A	2560x1600	PepolOnsSreet	30	150
		Traffic	30	150
B	1920x1080	Kimono	24	240
		ParkScene	24	240
		Cactus	50	500
		BQTerrace	60	600
		BasketBallDrive	50	500
C	832x480	BasketBallDrill	50	500
		BQMall	60	600
		RaceHorsesC	30	300
		PartyScene	50	500
D	416x240	BasketBallPass	50	500
		BQSquare	60	600
		BlowingBubbles	50	500
E	1280x720	FourPeople	60	600
		Johnny	60	600
		KristenAndSara	60	600

Table 7 Comparison of the proposed fast CU size decisions based on FSVM with [32–34] and [19]

Sequences	Proposed				Algorithm [32]				Algorithm [34]				Algorithm [19]				Algorithm [33]							
	BD-BR		BD-PSNR		BD-BR		BD-PSNR		BD-BR		BD-PSNR		BD-BR		BD-PSNR		BD-BR		BD-PSNR		BD-BR		BD-PSNR	
	(%)	(dB)	(%)	(dB)	(%)	(dB)	(%)	(dB)	(%)	(dB)	(%)	(dB)	(%)	(dB)	(%)	(dB)	(%)	(dB)	(%)	(dB)	(%)	(dB)	(%)	(dB)
PepeOnsStreet	-0.072	-0.012	-29.21	-0.02	0.44	-0.02	-0.062	-19.9	1.24	0.57	-0.062	-44.44	0.57	-0.03	-46.16	0.63	-0.03	-46.16	0.63	-0.03	-46.16	0.63	-0.03	-46.16
Traffic	-0.089	-0.010	-37.97	-0.05	0.92	-0.05	-0.051	-24.1	2.03	0.55	-0.051	-56.13	0.55	-0.04	-45.34	0.61	-0.04	-45.34	0.61	-0.04	-45.34	0.61	-0.04	-45.34
Average Class A	-0.081	-0.011	-33.59	-0.035	0.68	-0.035	-0.057	-22	1.64	0.56	-0.057	-50.29	0.56	-0.035	-45.75	0.62	-0.035	-45.75	0.62	-0.035	-45.75	0.62	-0.035	-45.75
Kimono	0.884	-0.018	-65.16	-0.02	0.75	-0.02	-0.068	-38.1	2.04	0.67	-0.068	-56.13	0.67	-0.02	-56.62	0.63	-0.02	-56.62	0.63	-0.02	-56.62	0.63	-0.02	-56.62
ParkScene	-0.095	-0.006	-37.56	-0.05	1.07	-0.05	-0.045	-31.6	1.46	0.50	-0.045	-47.82	0.50	-0.02	-41.91	0.57	-0.02	-41.91	0.57	-0.02	-41.91	0.57	-0.02	-41.91
Cactus	-0.125	-0.014	-51.05	-0.03	0.91	-0.03	-0.043	-29.0	1.77	0.45	-0.043	-48.44	0.45	-0.01	-42.35	0.54	-0.01	-42.35	0.54	-0.01	-42.35	0.54	-0.01	-42.35
BQTerrace	0.032	-0.007	-44.26	-0.05	0.83	-0.05	-0.037	-25.4	2.1	0.62	-0.037	-45.07	0.62	-0.04	-42.50	0.43	-0.04	-42.50	0.43	-0.04	-42.50	0.43	-0.04	-42.50
BasketBallDrive	-0.005	-0.046	-59.42	-0.02	0.89	-0.02	-0.054	-39.6	2.21	0.79	-0.054	-50.96	0.79	-0.02	-50.73	0.49	-0.02	-50.73	0.49	-0.02	-50.73	0.49	-0.02	-50.73
Average Class B	0.138	-0.018	-51.49	-0.034	0.89	-0.034	-0.049	-32.74	1.92	0.61	-0.049	-49.68	0.61	-0.022	-46.82	0.53	-0.022	-46.82	0.53	-0.022	-46.82	0.53	-0.022	-46.82
BasketBallDrill	-0.012	-0.018	-39.86	-0.04	0.90	-0.04	-0.068	-31.0	1.7	0.64	-0.068	-47.58	0.64	-0.05	-40.89	0.59	-0.05	-40.89	0.59	-0.05	-40.89	0.59	-0.05	-40.89
BQMall	-0.031	-0.011	-39.05	-	-	-	-	-	-	0.51	-	-	-	-0.04	-40.92	0.33	-0.04	-40.92	0.33	-0.04	-40.92	0.33	-0.04	-40.92
RaceHorsesC	0.012	-0.015	-42.01	-0.04	0.91	-0.04	-0.053	-31.0	1.21	0.60	-0.053	-33.37	0.60	-0.03	-38.75	0.53	-0.03	-38.75	0.53	-0.03	-38.75	0.53	-0.03	-38.75
PartyScene	-0.066	-0.008	-49.93	-	-	-	-	-	1.3	0.35	-	-	-	-0.02	-33.32	0.40	-0.02	-33.32	0.40	-0.02	-33.32	0.40	-0.02	-33.32
Average Class C	-0.024	-0.013	-42.71	-0.02	0.452	-0.02	-0.039	-15.5	1.05	0.53	-0.039	-29.40	0.53	-0.035	-38.47	0.46	-0.035	-38.47	0.46	-0.035	-38.47	0.46	-0.035	-38.47
BasketBallPass	0.425	-0.025	-42.58	-0.05	0.87	-0.05	-0.059	-31.0	1.19	0.52	-0.059	-36.63	0.52	-0.03	-46.26	0.43	-0.03	-46.26	0.43	-0.03	-46.26	0.43	-0.03	-46.26
BQSquare	0.016	-0.018	-44.44	-0.03	0.34	-0.03	-0.03	-14.5	0.84	0.10	-0.03	-30.67	0.10	-0.02	-47.41	0.32	-0.02	-47.41	0.32	-0.02	-47.41	0.32	-0.02	-47.41
BlowingBubbles	0.047	-0.022	-40.41	-	-	-	-	-	0.89	0.02	-0.036	-28.82	0.02	-0.01	-35.72	0.27	-0.01	-35.72	0.27	-0.01	-35.72	0.27	-0.01	-35.72
Average Class D	0.16	-0.02	-42.48	-0.03	0.40	-0.03	-0.04	-15.17	0.97	0.21	-0.04	-32.04	0.21	-0.02	-43.13	0.34	-0.02	-43.13	0.34	-0.02	-43.13	0.34	-0.02	-43.13
FourPeople	0.028	-0.020	-41.08	-0.07	1.23	-0.07	-0.097	-29.8	2.83	0.73	-0.097	-66.35	0.73	-0.05	-49.33	0.70	-0.05	-49.33	0.70	-0.05	-49.33	0.70	-0.05	-49.33
Johnny	0.179	-0.018	-54.01	-0.04	0.96	-0.04	-0.057	-46.7	2.48	1.04	-0.057	-70.93	1.04	-0.06	-61.52	1.01	-0.06	-61.52	1.01	-0.06	-61.52	1.01	-0.06	-61.52
KristenAndSara	0.010	-0.023	-52.61	-0.04	0.79	-0.04	-0.067	-43.7	2.29	1.00	-0.067	-68.23	1.00	-0.06	-59.80	0.70	-0.06	-59.80	0.70	-0.06	-59.80	0.70	-0.06	-59.80
Average Class E	0.07	-0.02	-49.23	-0.05	0.99	-0.05	-0.07	-40.07	2.53	0.92	-0.07	-68.50	0.92	-0.06	-56.88	0.80	-0.06	-56.88	0.80	-0.06	-56.88	0.80	-0.06	-56.88

Table 7 (continued)

Sequences	Proposed Algorithm			Algorithm [32]			Algorithm [34]			Algorithm [19]			Algorithm [33]		
	BD-BR (%)	BD-PSNR (dB)	ΔT (%)	BD-BR (%)	BD-PSNR (dB)	ΔT (%)	BD-BR (%)	BD-PSNR (dB)	ΔT (%)	BD-BR (%)	BD-PSNR (dB)	ΔT (%)	BD-BR (%)	BD-PSNR (dB)	ΔT (%)
Average	0.067	-0.017	-45.33	0.84	-0.04	-31.10	1.75	-0.05	-48.80	0.57	-0.03	-45.85	0.54	-0.03	-42.41

Table 8 Comparison of the proposed fast CU size decisions based on CNN in [14] and [16]

Sequences	Proposed			Algorithm			Algorithm		
	Algorithm			[14]			[16]		
	BD-BR	BD-PSNR	ΔT	BD-BR	BD-PSNR	ΔT	BD-BR	BD-PSNR	ΔT
	(%)	(dB)	(%)	(%)	(dB)	(%)	(%)	(dB)	(%)
PepolOnsSreet	1.99	-0.039	-63.86	1.17	-0.043	-66.86	2.16	-0.043	-62.73
Traffic	1.97	-0.042	-68.45	1.14	-0.044	-72.91	2.76	-0.029	-69.62
Average Class A	1.98	-0.04	-66.16	1.16	-0.04	-69.89	2.46	-0.04	-66.18
Kimono	1.43	-0.024	-86.41	1.17	-0.030	-74.52	2.15	-0.079	-68.64
ParkScene	1.65	-0.026	-67.46	1.41	-0.042	-72.61	2.13	-0.055	-69.79
Cactus	1.93	-0.023	-66.5	1.79	-0.028	-60.41	2.11	-0.065	-70.11
BQTerrace	1.32	-0.029	-62.89	1.68	-0.032	-65.02	1.47	-0.045	-71.23
BasketBallDrive	2.11	-0.040	-72.5	1.52	-0.031	-73.32	4.6	-0.027	-73.28
Average Class B	1.69	-0.03	-71.15	1.51	-0.03	-69.18	2.49	-0.05	-70.61
BasketBallDrill	1.73	-0.065	-60	1.52	-0.041	-76.22	2.97	-0.034	-61.1
BQMall	2.09	-0.023	-60.01	1.57	-0.027	-57.86	1.27	-0.048	-54.69
RaceHorses	1.27	0.005	-60	-	-	-	1.97	-0.108	-57.11
PartyScene	0.6	0.000	-50.59	0.32	-0.034	-54.02	0.5	-0.064	-50.67
Average Class C	1.42	-0.02	-57.65	0.85	-0.03	-56.87	1.02	-0.06	-53.67
BasketBallPass	2.17	0.008	-61.22	0.75	-0.043	-53.22	2.2	-0.037	-55.07
BQSquare	1.5	-0.047	-66.35	1.45	-0.024	-65.33	0.19	-0.062	-58.83
BlowingBubbles	0.19	-0.013	-52.8	0.34	-0.021	-52.05	0.68	-0.071	-47.11
Average Class D	1.29	-0.02	-60.12	0.85	-0.03	-56.87	1.02	-0.06	-53.67
FourPeople	2.35	-0.057	-68.57	1.61	-0.041	-65.69	3.66	-0.032	-67.49
Johnny	2.3	-0.040	-78.73	2.03	-0.032	72.01	3.6	-0.074	-75.7
KristenAndSara	2.16	-0.048	-76.32	1.54	-0.032	-77.57	3.21	-0.048	-74.01
Average Class E	2.27	-0.05	-74.54	1.73	-0.04	-71.76	3.49	-0.05	-72.40
Average	1.69	-0.030	-66.04	1.28	-0.04	-62.81	2.21	-0.05	-63.95

6.2 Comparison of proposed algorithms with the related works

Regarding the proposed algorithm based on FSVM, its performance is compared to SVM algorithms proposed by Jun et al. [32], Yun et al. [34], Zhang et al. [33] and Liu et al. [19] as summarized in Table 7. With regard to the complexity reduction, we noticed that the proposed FSVM method achieved 45.33% reduction on average, which is superior to 31.1% reduction obtained by [32] and 42.41% attained by [19]. However, the performances of the approaches used in [34] and [33] exceeded our results with a reduction of 48.8% and 45.85% respectively.

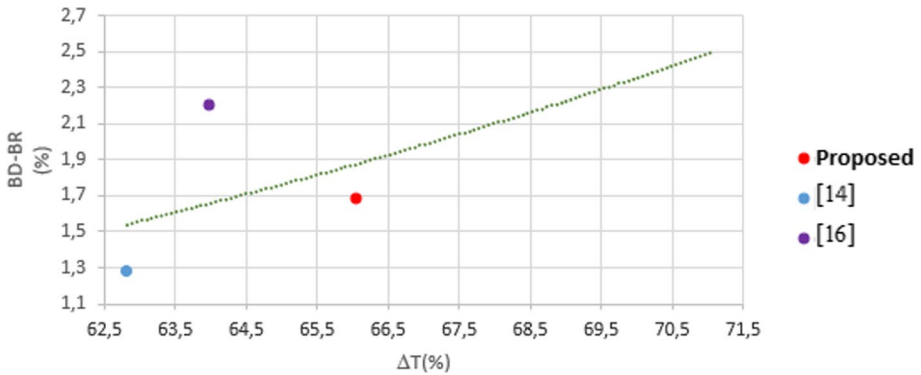


Fig. 10 Pareto analysis for the CNN based approaches

On the other hand, in addition to the complexity reduction, the RD performance is also considered as a critical evaluation metric. For all the listed approaches, insignificant video quality degradation was noticed. However, we can notice that, the BD-PSNR of our FSVM method averages -0.017 dB, which is better than -0.04 dB, -0.05 dB and -0.03 obtained respectively in [32, 34] and [19] approaches. On the other hand, the BD-BR of our method is averagely 0.067%, which outperforms 0.84% , 1.75%, 0.57% and 0.54% obtained respectively with [34], 0.57% of [19] and [33]. Consequently, our proposed algorithm outperforms the approaches [32–34] and [19] in terms of RD performance. Even if [34] and [33] approaches achieved a better reduction complexity performance, this improvement is widely penalized with an important compression efficiency degradation compared to our results. Since the coding efficiency is the main goal of video compression, we can consider that our FSVM scheme achieved the best tradeoff between the compression efficiency and the complexity reduction performances when optimizing the HEVC complexity at all Intra configuration. The experimental results demonstrate that tiny ground truth is crucial to the efficient encoding decision prediction when using FSVM based prediction models.

Our proposed CNN based approach is compared with the algorithms used by Kuanar et al. [14] and Li et al. [16] to evaluate its performance in terms of complexity reduction and coding efficiency when observing its impact on video quality. According to Table 8, our proposed algorithm reduces the average computational complexity up to 66.04% with an RD performance loss up to 1.69% in terms of BD-BR and -0.03 dB in terms of BD-PSNR. The related works algorithms [14] and [16] achieved respectively 63.59% and 62.81% in terms of computational complexity reduction and a compression efficiency

Table 9 Results of FSVM model compared to CNN methods

Class	FSVM			CNN		
	BD-BR(%)	BD-PSNR(db)	ΔT (%)	BD-BR(%)	BD-PSNR(db)	ΔT (%)
A	-0.081	-0.011	-33.59	1.98	-0.041	-66.16
B	0.138	-0.018	-51.49	1.69	-0.028	-71.15
C	-0.024	-0.013	-42.71	1.42	-0.021	-57.65
D	0.163	-0.022	-42.47	1.29	-0.017	-60.12
E	0.072	-0.020	-49.23	2.27	-0.048	-74.54

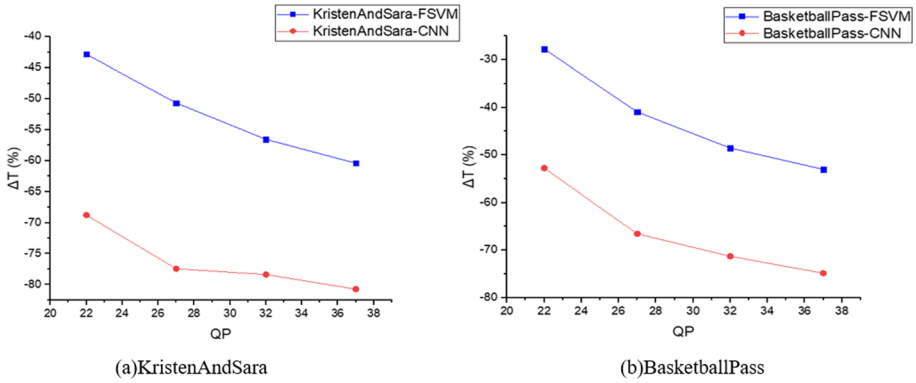


Fig. 11 QP impact on complexity reduction performance for FSVM and CNN approaches

degradation up to 1.28% and 2.21% respectively, while maintaining the same video quality performance.

To complement this study, the Pareto fronts [7] have been conducted in Figure 10 to confirm that, undeniably, the tradeoff between the complexity reduction and coding efficiency performances of our proposed CNN method are better than approach [14] that provides the least bit-rate performance degradation.

Fig. 12 The FSVM classifier’s training complexity

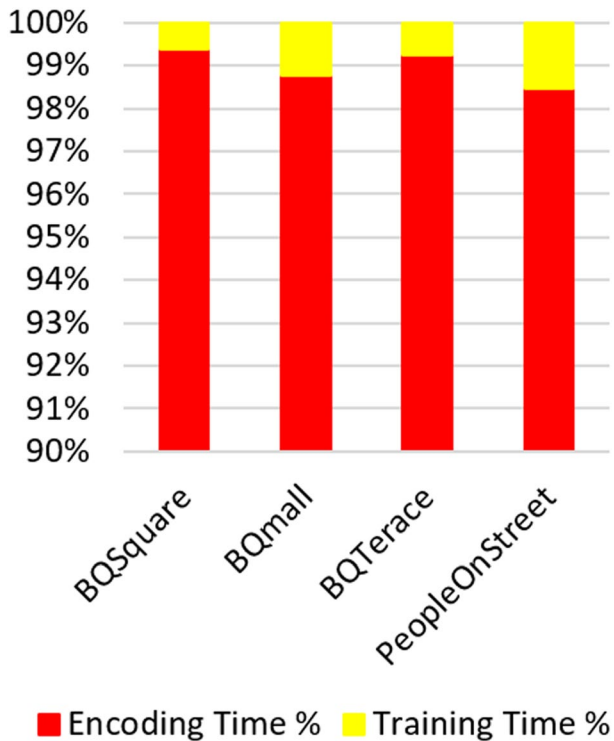
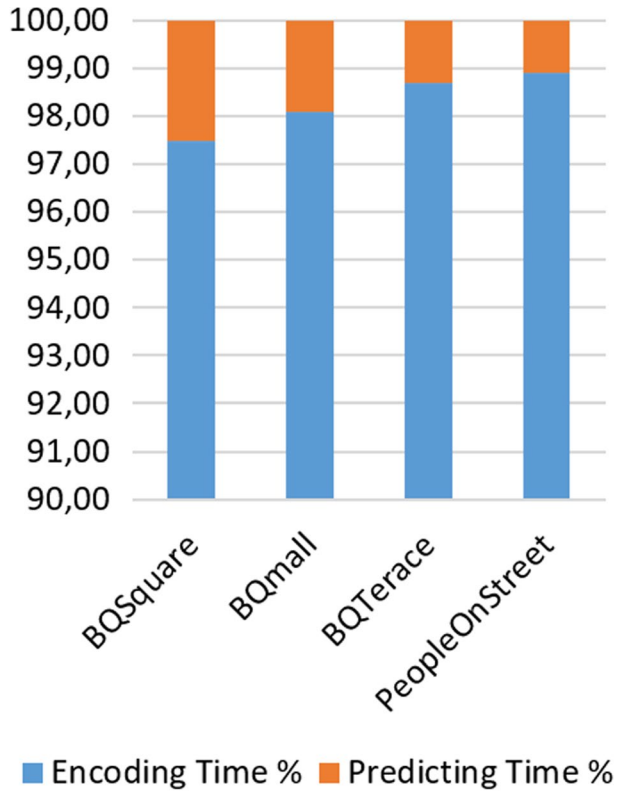


Fig. 13 The CNN prediction process complexity



The best tradeoff between complexity reduction and compression efficiency degradation is noted for the Kimono video sequence. This class B video saves 86% of the encoding time with an increase in BD-BR of 1.41% and a little decrease in BD-PSNR. An interesting tradeoff is also obtained with class B cactus video. We can conclude that despite the high motion activity noted in class B videos needing to be coded with smaller details, the proposed CNN approach is very efficient to correctly predict small CU size with less time consumption.

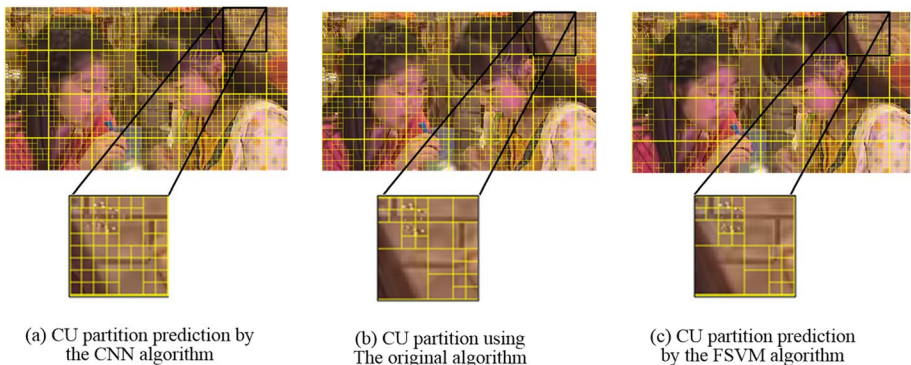


Fig. 14 CU prediction results for Frame 50 of BlowingBubbles video at QP37

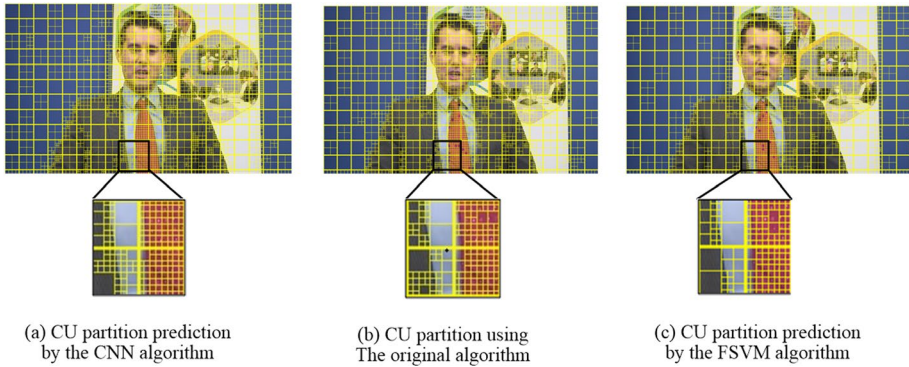


Fig. 15 CU prediction results for the Frame 50 of Johnny video at QP27

6.3 Performances comparison between proposed CNN and FSVM algorithms

The performances comparison between proposed CNN and FSVM algorithms to reduce CU partitioning complexity for HEVC encoder at all Intra configuration is illustrated in Table 9. We can clearly notice that CNN based approach performance overpasses FSVM based one for different class resolutions. However, because of the use of all intra configuration, the main goal of our research is to optimize the CU prediction module, while maintaining HEVC compression efficiency performance. Thus, FSVM with a tiny coding efficiency degradation compared to CNN, can be considered as the approach that performs the best tradeoff between the compression efficiency and the complexity reduction for HEVC all intra configuration. Indeed, the FSVM coding efficiency improvement in class A and C exceeded the original HM test model performance when performing more than 33% complexity reduction.

QP impact on complexity reduction performance for FSVM and CNN approaches is illustrated on Figure 11. We can observe that the reduction complexity of the proposed CNN based implementation and FSVM approaches increases along with the increased QP. It is mainly because the increased QP leads to larger CUs selection, which results in a running time reduction.

To test the additional classifier’s training complexity of the proposed FSVM method, and the prediction process complexity of the proposed CNN approach, the percentages of time consumption are reported in Figure 12. Four sequences, BQSquare (416 x 240), BQMall (832 x 480), BQTerrace (1920 x 1080), and PeopleOnStreet (2560x1600), are adopted and encoded under all Intra configuration at QP 37 with different resolutions and contents. The results reveal that FSVM classifier’s training time consumption is around 1%. For larger resolution sequences, more training samples are performed within

Table 10 Comparison of different methods with respect to HEVC reference software

	Works	BD-BR(%)	BD-PSNR(dB)	ΔT(%)
Mode predictions	[8]	0.1	-	-14.4
	[11]	1.92	-0.11	-40.90
CU decision	Proposed CNN	1.69	-0.030	-66.04
	Proposed FSVM	0.067	-0.017	-45.33

a fixed number of training frames. Therefore, the additional complexity of the proposed FSVM classifier's training has no degrading impact on the overall HEVC complexity improvement.

The time required to predict CU size decisions within 100 frames using the proposed CNN algorithm takes only 1.7% of the overall optimized HEVC encoding time as shown in Figure 13. In conclusion, the additional complexity of the proposed CNN prediction process does not affect the overall HEVC complexity improvement.

Figure 14 and Figure 15 expose the results of the predicted CUs on frame 50 at QP 37 respectively for *BlowingBubbles* and *johnny* video sequences. Compared to the ground truths results, we can perceptually note that the proposed FSVM algorithm achieves more truthful CUs predictions than the CNN proposed algorithm, which explains the compression performance degradation engendered with the CNN approach.

The comparison between performances of methods acting on HEVC intra-mode prediction process [8] and [11], and proposed methods acting on HEVC CU size decision module are illustrated on Table 10. From the overall performance assessment, we can infer that the proposed methods achieve better optimization results: 66% and 45% for CNN and FSVM approaches respectively, compared to 14% and 40% obtained with [8] and [11] respectively. This observation proves that we have made the best choice to select the CU size decision module, as a first optimization step of the HEVC encoder at all intra configuration.

7 Conclusion and future work

In this paper, two machine learning-based methods are proposed to perform fast CU partition for HEVC at all intra configuration. Since the CU partition is modeled as a multi-classification problem, a cascade of three binary classifiers is adopted in the FSVM based method and adequate features are selected to allow an interesting accuracy. With an automatically feature extraction process, a second approach is proposed using three deep CNN models learned to predict the optimal CU partition based on an efficient map representation and sufficient training data.

The experimental complexity reduction results showed that the proposed deep CNN approach widely outperformed the proposed FSVM one with 66.4% against 45.33%. However, the FSVM with only 0.067% loss in coding efficiency compared to 1.69% degradation engendered with the CNN is the most suitable solution to reduce the HEVC CU partitioning module complexity at all intra configuration.

We would assert that our work is a step that may be taken further to enhance the HEVC performance. It would be interesting to combine the resulting optimal CU decision module with the additional optimized computational HEVC modules such as the intra-mode prediction process. This research work is ongoing, and the results will be presented in future publications.

References

1. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado GS, Davis A, Dean J, Devin M, et al (2016) Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint [arXiv:1603.04467](https://arxiv.org/abs/1603.04467)

2. Amna M, Imen W, Ezahra SF (2021) Lenet5-based approach for fast intra coding. In 2020 10th International Symposium on Signal, Image, Video and Communications (ISIVC), pp. 1–4
3. Bjontegaard G (2001) Calculation of average psnr differences between rd-curves. VCEG-M33
4. Bouaafia R, Khemiri S, Sayadi FE, Atri M (2020) Fast cu partition-based machine learning approach for reducing hevc complexity. *J Real Time Image Process* 17(1):185–196
5. Chang CC, Lin CJ (2011) Libsvm: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)* 2(3):1–27
6. Correa G, Assuncao P, Agostini L, L.A. da SilvaCruz, (2012) Performance and computational complexity assessment of highefficiency video encoders. *IEEE Transactions on Circuits and Systems for Video Technology* 22(12):1899–1909
7. Fernandez DG, Alberto A, Del B, Botella G, Meyer-Baese U, Meyer-Baese A, Grecos C (2017) Information fusion based techniques for hevc. In *Real-Time Image and Video Processing* 10223:102230M
8. Fini MR, Zargari F (2019) Two stage fast mode decision algorithm for intra prediction. *HEVC. Multimedia Tools and Applications* 75(13):7541–7558
9. Grellert M, Zatt B, Bampi S, da Silva Cruz LA (2018) Fast coding unit partition decision for HEVC using support vector machines. *Trans Circuits Syst Video Technol.* 29(6):1741–1753
10. Hassan M, Shanableh T (2019) Predicting split decisions of coding units in hevc video compression using machine learning techniques. *Multimedia Tools and Applications* 78(23):32735–32754
11. Hosseini E, Pakdaman F, Hashemi MR, Ghanbari M (2019) A computationally scalable fast intra coding scheme for hevc video encoder. *Multimedia Tools and Applications* 78(9):11607–11630
12. JCT-VC (2019) Hm software. <https://hevc.hhi.fraunhofer.de/>. [Online; Accessed 15 Apr 2019]
13. Jiang W, Ma H, Chen Y (2012) Gradient based fast mode decision algorithm for intra prediction in hevc. In 2012 2nd international conference on consumer electronics, communications and networks (CECNet), pp. 1836–1840
14. Kuanar S, Rao KR, Conly C (2018) Fast mode decision in hevc intra prediction, using region wise cnn feature classification. In 2018 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), pp. 1–4
15. Lan C, Xu J, Sullivan GJ, Wu F (2012) Intra transform skipping, document jctvc-i0408. Joint Collaborative Team on Video Coding (JCT-VC)
16. Li T, Xu M, Deng X (2017) A deep convolutional neural network approach for complexity reduction on intra-mode hevc. In 2017 IEEE International Conference on Multimedia and Expo (ICME), pp. 1255–1260
17. LI Z, Zhang Y, Li B (2012) Gradient-based fast decision for intra prediction in HEVC. In *Vis Commun Image Process.* pages 1–6
18. Lin CF, Wang SD (2002) Fuzzy support vector machines. *IEEE transactions on neural networks* 13(2):464–471
19. Liu X, Li Y, Liu D, Wang LT, Yang P (2017) An adaptive cu size decision algorithm for hevc intra prediction based on complexity classification using machine learning. *IEEE Transactions on Circuits and Systems for Video Technology* 29(1):144–155
20. Liu Z, Yu X, Gao Y, Chen S, Ji X, Wang D (2016) Cu partition mode decision for hevc hardware intra encoder using convolution neural network. *IEEE Transactions on Image Processing* 25(11):5088–5103
21. Mallikarachi T, Fernando A, Arachchi HK (2014) Efficient coding unit size selection based on texture analysis for hevc intra prediction. In 2014 IEEE International Conference on Multimedia and Expo (ICME), pp. 1–6
22. Moraes D, Wainer J, Rocha A (2016) Low false positive learning with support vector machines. *Journal of Visual Communication and Image Representation* 38:340–350
23. Shen X, YU L (2013) A fast hevc inter cu selection method based on pyramid motion divergence. *EURASIP journal on image and video processing* 2013(1):1–11
24. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15(1):1929–1958
25. Sullivan GJ, Ohm JR, Han WJ, Wiegand T (2012) Overview of the high efficiency video coding (HEVC) standard. *IEEE Trans Circuits Syst Video Technol* 22(12):1649–1668
26. Tsai AC, Wang JF, Lin WG, Yang JF (2007) A simple and robust direction detection algorithm for fast h. 264 intra prediction. In 2007 IEEE International Conference on Multimedia and Expo, pp 1587–1590
27. Wang C, Yu L, Wang S (2018) Accelerate cu partition in hevc using large-scale convolutional neural network. arXiv preprint [arXiv:1809.08617](https://arxiv.org/abs/1809.08617)
28. Wiegand T, Sullivan GJ, Bjontegaard G, Luthra A (2003) Overview of the h. 264/avc video coding standard. *IEEE Transactions on circuits and systems for video technology* 13(7):560–576

29. Xiph.org (2019) Xiph.org video test media. <https://media.xiph.org/video/derf>. [Online; Accessed 15 Apr 2019]
30. Xiong J, Li H, Wu Q, Meng F (2013) A fast hevc inter cu selection method based on pyramid motion divergence. *IEEE transactions on multimedia* 16(2):559–564
31. Xu M, Li T, Wang Z, Deng X, Yang Z, Guan R (2018) Reducing complexity of hevc: A deep learning approach. *IEEE Transactions on Image Processing* 27(10):5044–5059
32. Yin Y, Yang X, Lin J, Chen Y, Fang R (2018) A fast block partitioning algorithm based on svm for hevc intra coding. In *Proceedings of the 2018 the 2nd International Conference on Video and Image Processing*, pp. 176–181
33. Zhang T, Sun MT, Zhao D, Gao W (2016) Fast intra-mode and cu size decision for hevc. *IEEE Transactions on Circuits and Systems for Video Technology* 27(8):1714–1726
34. Zhang Y, Kwong A, Wang X, Yuan H, Pan Z, Xu L (2015) Machine learning-based coding unit depth decisions for flexible complexity allocation in high efficiency video coding. *IEEE Transactions on Image Processing* 24(7):2225–2238
35. Zhu L, Zhang Y, Pan Z, Wang R, Kwong S, Peng Z (2017) Binary and multi-class learning based low complexity optimization for hevc encoding. *IEEE Transactions on Broadcasting* 63(3):547–561