Fast Mode and Partition Decision Using Machine Learning for Intra-Frame Coding in HEVC Screen Content Coding Extension

Fanyi Duanmu, Zhan Ma, and Yao Wang, *Fellow, IEEE*

***Abstract*— This paper presents a fast mode and partition decision framework for screen content coding (SCC) based on machine learning. Extensive statistical studies and complexity evaluations are conducted to explore the distribution of different coding modes and their complexities. The proposed encoder scheme is designed based on the results from these studies. Firstly, a CU is classified as either a natural image block (NIB) or a screen content block (SCB). An SCB only goes through SCC modes at the current CU level. An NIB is further classified as "partitioned" or "non-partitioned". The partitioned block will bypass current level intra modes. The non-partitioned block is classified as "directional" or "non-directional" and only goes through a subset of intra candidates. Decision tree classifiers are designed with chosen features that can distinguish different types of blocks. Furthermore, additional mode/partition checking is terminated once the current mode coding rate is lower than a statistics-based threshold. Compared with HEVC-SCC reference software, our proposed fast encoder can balance the encoding efficiency and complexity by adjusting decision confidence thresholds and rate thresholds. Under all-intra configurations, our "rate-distortion preserving" setting can achieve 40% complexity reduction with only 1.46% BD-loss. Our "complexity-reduction boosting" setting can achieve 52% complexity reduction with 3.65% BD-loss.**

***Index Terms*— Decision tree, fast mode decision, high efficiency video coding, machine learning, screen content coding.**

I. Introduction

A. Motivation

SCREEN content (SC) videos have become more and more popular due to the recent advances in mobile technologies and cloud applications, such as shared screen collaboration, remote desktop interfacing, cloud gaming, wireless display, animation storage and streaming, online education, etc. These emerging cloud-based applications and market demands create an urgent need for efficient compression and low-latency delivery of screen content videos, especially to support the incoming 4K or even higher resolution.

Manuscript received March 8, 2016; revised June 7, 2016 and July 16, 2016; accepted July 18, 2016. Date of publication August 12, 2016; date of current version December 9, 2016. The work of Z. Ma was supported in part by National Science Foundation for Young Scholar of Jiangsu Province, China (Grant BK20140610) and the National Science Foundation of China (Grant 61571215). This paper was recommended by Guest Editor W.-H. Peng.

F. Duanmu and Y. Wang are with the Department of Electrical and Computer Engineering, Tandon School of Engineering, New York University, Brooklyn, NY 11201 USA (e-mail: fanyi.duanmu@nyu.edu; yaowang@nyu.edu).

Z. Ma is with Nanjing University, Nanjing 210093, China (e-mail: mazhan@nju.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JETCAS.2016.2597698

Recognizing the need for industrial standardization of screen content coding (SCC), the ISO/IEC Moving Picture Expert Group and ITU-T Video Coding Experts Group, also referred as "Joint Collaborative Team on Video Coding" (JCT-VC), has launched the standardization of SCC extension [1] on top of the latest video standard—High Efficiency Video Coding (HEVC) [2] since January 2014. Research efforts from both academia and industry have been put together, to better explore and understand the unique SC signal characteristics and develop efficient SCC algorithms.

The official JCTVC Screen Content Model software Ver. 4 (SCM-4.0) [3] can achieve over 55% BD-rate saving over HEVC Range Extension (REext) [1] for computer-generated contents. This significant compression improvement is attained from four major coding tools beyond HEVC, known as "Intra Block Copy" (IBC) [4], [5], "Palette Coding Mode" (PLT) [6], "Adaptive Color Transform" (ACT) [7] and "Adaptive Motion Compensation Precision" (AMCP) [8], [9], respectively.

However, the intrinsic HEVC recursive partitioning scheme and the additional SCC tools impose significant computational burden on the encoder, primarily during the seeking of optimal combinations of CU partitions and CU modes, as summarized into the following two fundamental problems:

Mode Decision: given current CU, which mode (among 35 Intra modes, PLT mode and IBC mode) should be chosen to optimally minimize the Rate-Distortion (RD) cost?

Partition Decision: given current CU, should it be further partitioned into smaller CUs for a better RD performance?

In existing SCM software, an exhaustive search method is employed to resolve these two problems by examining every possible mode for the current CU and comparing the RD cost of current CU using the best mode and the sum of RD costs for all its recursive sub-CUs (i.e., the smaller CUs that the current CU splits into), each encoded with optimal mode and partition. Even though there are some fast termination schemes (e.g., IBC Skip fast termination [10]) and fast search algorithms (e.g., hash-based IBC/Inter search [11], [12]) included in the SCM-4.0 software, existing exhaustive mode/partition search scheme is still not practical for real-time applications. Thus, it is critically important to develop fast algorithms for making the partition or mode decisions while preserving SCC efficiency.

Although there have been substantial prior research efforts in fast mode and partition decisions for the HEVC encoder, to our best knowledge, very few prior works have been

published for fast mode and partition decisions for HEVC SCC extension. The IBC and PLT modes introduced in the SCC extension make fast decision problems extremely challenging, as will be demonstrated in Section II-D. Our proposed solution tackles this challenge up front, by classifying a CU into either a Natural Image Block (NIB) or a Screen Content Block (SCB), and examines only the traditional Intra modes for an NIB, and examines only the IBC and PLT modes for an SCB. To account for the possibility that an NIB may have a good match in the previously-coded area, which mostly happens in smaller CUs, we also invoke IBC skip and merge modes for NIBs at CU8 level, to achieve a good balance between the coding efficiency and complexity.

B. Previous Works

There are many prior works on HEVC encoder accelerations.

These papers can be categorized into the following categories:

Category 1: Mode Reduction. A gradient-based fast mode decision framework was proposed in [13], which bases on CU directional histogram analysis to reduce the number of intra candidates before mode selection. A reported 20% complexity reduction over HM-4.0 is achieved under this scheme with negligible coding performance loss for Intra-frame coding. Another fast intra mode decision algorithm was proposed in [14], which exploits the directional information of neighboring blocks to reduce the Intra candidates of the current CU. Up to 28% complexity reduction is reported over HM-1.0 with insignificant coding performance loss for Intra-frame coding. The HM test model software adopted [15] to reduce Intra-frame coding candidates. Firstly, a rough mode decision (RMD) is performed using Hadamard cost to choose fewer candidates out of 35. Then the extra most probable modes (MPMs) derived from spatial neighbors will be added to the previous candidate set if they are not yet included.

Category 2: Cost Replacement. An entropy-based fast Coding Tree Unit partition algorithm was proposed in [16], which replaces heavy Rate-Distortion optimization (RDO) calculation by Shannon entropy calculation. A 60% complexity reduction is reported using this algorithm with a BD-rate loss of 3.8% for Intra-frame coding. In [15], Hadamard cost is used for Intra RMD without fully formulating the RD cost. This approach significantly reduces the intra coding complexity.

Category 3: Fast Partition Termination. A fast CU splitting decision scheme was proposed in [17], using weighted SVM decision for early CU partition termination for both Intra-frame and Inter-frame coding. A complexity reduction of over 40% is reported over HM-6.0. Another fast termination algorithm was proposed in [18], using texture complexity of neighboring blocks to eliminate unnecessary partition of the current CU. A 23% encoder speed-up on average is reported over HM-9.0 for Intra-frame coding. Another work by Zhang and Ma [19] includes a set of early termination criteria for HEVC intra coding based on experimental observation and simulation results. To determine the splitting decision, encoder will do a one-level RD evaluation by comparing current CU Hadamard cost with the combined Hardmard cost of four sub-CUs without further splitting. Zhang and Ma further proposed an improved three-step fast HEVC Intra coding algorithm in [20]. At the RMD step, a 2:1 down-sampled Hadamard transform is used to approximate the encoding cost followed by a progressive mode refinement and early termination verification. It reports an averaged 38% complexity reduction over HM-6.0 with 2.9% BD-rate loss for Intra-frame coding.

Category 4: Fast Search Algorithm. A number of fast motion estimation (ME) algorithms have been proposed in the past years, including multi-step search [21], [22], diamond search [23], cross-diamond search [24], hexagon search [25], etc. These algorithms follow different search patterns to reduce the number of search points for inter-frame coding. In HEVC Test Model software (HM), Enhanced Predictive Zonal Search (EPZS) [26] is incorporated to reduce encoder complexity, in which prediction is continuously refined within local search using a small diamond or square pattern and the updated best vector becomes the new search center.

To conclude, these prior contributions were proposed for natural video coding without considering the unique signal properties of screen contents. Different from camera-captured natural videos, computer-generated screen contents (such as text, icon, curves, etc.) typically contain fewer distinct colors, sharper edges, repetitive graphical patterns, less complicated local textures and irregular motion fields. Besides, these prior contributions did not consider the newly-introduced SCC tools. PLT and IBC mode selections are dependent on the SC patterns and colors in the previously-coded area. Conventional local information based fast algorithms and homogeneity-based fast partition algorithms cannot apply to SCC directly.

There are a few recent works on SCC fast encoding. For instance, Li and Xu presented a fast algorithm for AMCP [27] to quickly determine the frame type (namely, SC image or Natural image), based on the percentage analysis of "smooth blocks," "collocated blocks," "matched blocks," and "other blocks" (i.e., the blocks that do not belong to the previous three categories). For similar encoder complexity, an averaged 7.7% BD-Rate improvement is reported over SCM-2.0 for Inter-frame coding. Kwon and Budagavi proposed a fast IBC search algorithm [28], by imposing restrictions on IBC search range, search directions and motion compensation precision. There are also several works on hash-based fast search algorithms for IBC mode and inter mode coding [11], [12], [29]. Tsang, Chan and Siu proposed a Simple Intra Prediction (SIP) scheme [30] to bypass RMD and RDO processing for smooth SCC regions, whose CU boundary samples are exactly the same. Up to 26.7% peak complexity reduction is reported over SCM-2.0 with marginal video quality degradation for Intra-frame coding. Lee et al. proposed a fast Transform Skip Mode Decision scheme for SCC [31], by enforcing IBC block with zero coded block flag (CBF) to be encoded with transform skip mode. A 3% encoding speed-up is reported over SCM-2.0. Zhang, Guo, and Bai proposed a Fast Intra Partition Algorithm [32] for SCC, using CU entropy and CU coding bits to determine CU partition decision for Intra-frame coding. In a recent work [33] proposed by Zhang and Ma, temporal

CU depth correlations are exploited and adaptive block search step size is incorporated on top. Average complexity reductions of 39% and 35% are reported for lossy and lossless Intra-frame coding, respectively. In our previous work [34], we propose a neural network (NN) based CU fast partition decision scheme using CU low-level statistical features (such as sub-CU consistency, CU variance, CU color number, etc.) as NN inputs to calculate CU partition soft-decision. A 37% complexity reduction is achieved under this scheme with only 3% BD-loss for Intra-frame coding. The current work significantly extends our previous work by considering the fast decision of both mode and partition jointly, and is able to achieve more significant complexity reduction at lower BD loss.

C. Our Contributions

Unlike most of the aforementioned SCC fast encoder algorithms, which consider how to speed up a particular mode, our work focuses on fast mode and partition decisions. This framework is applied without changing the implementations of Intra, IBC and PLT modes. Namely, the previous SCC fast algorithms on each individual tool could be easily incorporated into our proposed framework for additional encoder speed-up. Our contributions in this paper are two folds.

Firstly, we conducted extensive statistical investigations and complexity studies on SCC. Such information enables us to explore the relationship between SC image characteristics and SCM mode selection behaviors and preferences.

Secondly, we propose a Machine Learning based Fast SCC (ML-FSCC) scheme for fast CU mode and partition decision. The proposed scheme includes three classifiers designed to either reduce the mode candidates to be checked or to make fast partition decisions. Our experimental results demonstrate that the proposed fast algorithm can provide significant complexity reduction while simultaneously preserving SCC efficiency.

The sequel of the paper is structured as follows. Section II briefly reviews SCM coding structure, SCC new coding tools and discusses about the major technical challenges for SCC fast partition/mode decisions beyond conventional HEVC scheme. Section III provides statistical studies on SCM mode selection behaviors and complexity distribution. Section IV presents our proposed ML-FSCC coding framework in details. Section V explains our feature selection, classifier selection, parameter training methodologies and the classifiers design. In Section VI, experimental results and evaluations are presented and this paper concludes in Section VII with our tentative future work summarized.

II. HEVC Screen Content Model: A Brief Review

SCM is the JCTVC official test model software for SCC extension. This software is developed upon HEVC-REext and supports YUV4:4:4, YUV4:2:0, and RGB4:4:4 sampling formats. Beyond HEVC, new SCC modes (e.g., IBC, PLT, etc.) are introduced to improve SCC efficiency.

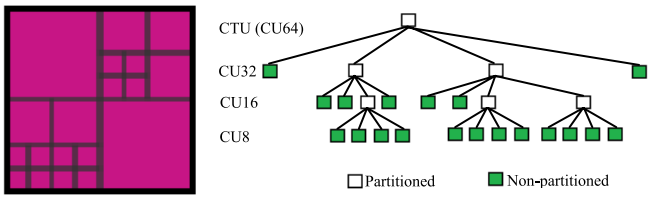

Fig. 1. SCM CU hierachitical quadtree partitioning structure.

A. SCM Mode and Partition Decisions

SCM shares the same flexible quadtree block partitioning scheme as HEVC, which enables the use of CUs, Prediction Units (PUs) and Transform Units (TUs) to adapt to diverse picture contents. CU is the basic unit for mode decision and is always in square shape. The Coding Tree Unit (CTU) is of 64x64 pixels by default.

At encoder, pictures are divided into non-overlapping CTUs and each CTU can be further divided into four equal-sized smaller CUs recursively, until the maximum hierarchical depth is reached, as shown in Fig. 1. At each CU-level, to determine the optimal encoding parameters (e.g., partition decision, mode decision and tool usages, etc.), an exhaustive search method is currently employed by comparing RD costs using different coding modes and comparing the minimum RD cost at current CU level against the sum of RD costs of its sub-CUs (each using best mode and partition). For the rest of this paper, we will use "CU64" (i.e., CTU), "CU32,", "CU16," and "CU8" to denote CUs at different hierachical depths.

B. SCM New Coding Tools beyond HEVC

Beyond HEVC, SCM adopted four major encoding tools to compress SC more efficiently.

Intra Block Copy [4], [5] is an intra-frame version of the block motion compensation scheme. To compress the current CU, the encoder will look back into previously-coded area (either in restricted area or globally) and find the best matching block. If chosen, a "Block Vector" (BV) will be signaled to indicate the relative spatial offset between the best matching block and the current CU.

Palette Mode [6] encodes current CU as a combination of a color table and an index map. Color table stores representative color "triplets" of RGB or YUV. Then the original pixel block is translated into a corresponding index map indicating which color entry in the color table is used for each pixel location.

Adaptive Color Transform [7] converts residual signal from original RGB or YUV color space onto YCoCg color space. It decorrelates the color components, reduces the residual signal energy and therefore improves the coding performance.

Adaptive Motion Compensation Precision [8], [9] analyzes inter-frame characteristics and categorizes current frame into either natural video or SC video. For SC video, integer-pixel precision is applied for motion estimation. For natural video, sub-pixel precision is applied.

C. SCM-4.0 IBC and Inter Unification

Beyond previous SCM releases, IBC mode and Inter mode are unified in SCM-4.0 release. Namely, IBC mode is treated as a special Inter mode with reference frame restricted to

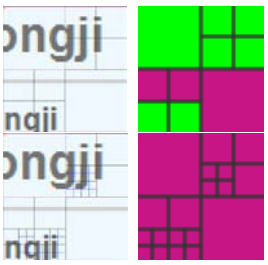

Fig. 2. CTU partition decision comparison between SCM-4.0 (top) and HEVC (bottom). "Green" blocks are coded in PLT mode and "Magenta" blocks are coded in HEVC-Intra mode.

current frame and reference area restricted to previously-encoded area in the current frame. This unification improves SCM encoding from the following perspectives.

1) Workflow harmonization for hardware re-utilization.

2) Inter-frame CU encoding flexibility. Inter-frame CU can have its PUs copied from both the current frame and temporal frame simultaneously.

3) Inter-frame coding schemes generalization for IBC blocks, such as "Advanced Motion Vector Prediction" (AMVP) [35], "Merge/Skip mode" [36], etc. Later in this paper, we will use "IBC-Merge," "IBC-Skip," and "IBC-Inter" to address the terminology difference from conventional HEVC.

4) Inter-frame fast algorithms transfer onto Intra-frames. For instance, when IBC finds a "Skip" candidate, the encoder will immediately terminate all following RDOs.

D. SCM-4.0 Fast Mode and Partition Decision Challenges

The selections of PLT mode and IBC mode are both highly dependent on the graphic patterns and image colors that appeared previously. This "historical dependency" makes fast CU partition and mode decision problems much more complicated and challenging than for conventional HEVC encoder. For IBC mode, depending on whether similar pattern appeared previously, encoding costs of the same CU pattern but at different locations may vary significantly. Similarly, for palette mode, two color tables are created. One is used for current CU and the other (also known as "palette predictor") is served as a dynamic lookup table storing the historical colors frequently used. For the same CU pattern but at different locations, depending on whether similar colors appeared previously and how frequent these colors are, the PLT encoding costs may vary significantly.

Because of the aforementioned historical dependency, CU homogeneity and signal entropy do not suffice to make fast CU partition decision. Furthermore, new coding tools make "inhomogeneous" CUs possible to be encoded as a whole block without splitting. As shown in Fig. 2, several 16x16 textual CUs in the top row are directly coded using PLT mode (marked in green) without further splitting into smaller 8x8 intra CUs (marked in magenta) in the bottom row.

To summarize, traditional HEVC fast mode and partition algorithms cannot be effectively applied to SCC, due to the unique SC signal properties and the introductions of PLT and IBC modes. Furthermore, the additional encoding options (e.g., ACT, Transform Skip, index map scan order, etc.) also complicate SCC fast decisions.

Fig. 3. Sample frames from JCT-VC SCC sequences.

TABLE I
SAMPLE FRAME SELECTION

Sequence	Sample Frame Index
Map	0, 484, 73, 151, 112, 100, 61, 65, 102, 63
WebBrowsing	0, 286, 100, 67, 254, 8, 34, 144, 120, 188
Programming	0, 73, 568, 598, 43, 70, 45, 38, 244, 48
SlideShow	0, 3, 27, 474, 170, 169, 171, 167, 168, 266
Robot	0, 8, 9, 2, 12, 130, 10, 11, 131, 132
FlyingGraphics	0, 278, 144, 111, 141, 84, 100, 73, 171, 240
Desktop	0, 463, 501, 422, 480, 479, 508, 128, 30, 31
Console	0, 418, 130, 599, 439, 137, 509, 465, 419, 429
Basketball	482, 540, 538, 376, 534, 504, 409, 479, 493, 439
MissionControlClip2	209, 255, 211, 154, 158, 206, 259, 161, 260, 151
MissionControlClip3	0, 478, 146, 481, 479, 475, 362, 483, 474, 365

III. SCM-4.0 MODE AND PARTITION STATISTICAL ANALYSIS

To better understand the unique SC signal characteristics and SCM-4.0 encoder complexity distribution and mode selection behaviors, we conducted several statistical studies, as follows.

A. Test Sample Preparation

Our statistical study is based on SCM-4.0 encoding results over 11 standard sequences jointly selected by the experts from JCTVC. These sequences cover "Text & Graphics" (TG), "Mixed Contents" (MC) and "Animation" (AM) categories and include the most typical screen content applications, such as "Desktop," "Console," "Map," "WebBrowsing," "SlideShow," etc., as shown in Fig. 3. The sequences and coding parameters are described in JCTVC SCC Common Testing Conditions (CTC) [37].

For simplicity, to avoid similar frames and duplicate CU samples in the training set, we extract 10 sample frames from each sequence whose pixel-wise temporal difference against the previous frames are the largest over Y component. The selected frames (in Table I) are coded using All-Intra (AI) configurations under QP22, QP27, QP32, and QP37. The mode selection and partition decision labels derived from SCM-4.0 encoder are used as ground-truth data for our statistical study and machine learning in the following sections.

B. SCM-4.0 Mode and Partition Distribution

Firstly, we conduct a distribution analysis investigating the relationships between CU mode, CU partition and CU size.

TABLE II
SCM-4.0 PARTITION STATISTICS (QP = 22)

CU Width	Partition #	Partition %	Non-Partition #	Non-Partition %
64	41184	86.16%	6616	13.84%
32	105238	70.02%	45052	29.98%
16	303720	59.98%	202680	40.02%

TABLE III
SCM-4.0 MODE STATISTICS AMONG NON-PARTITIONED BLOCKS (QP = 22)

CU Width	IBC Merge	IBC Skip	IBC Inter	Intra	PLT
64	1.45%	1.10%	Disabled	11.29%	Disabled
32	1.47%	12.46%	2.13%	8.15%	5.76%
16	1.31%	15.70%	5.44%	12.18%	5.40%
8	5.92%	19.44%	23.22%	44.25%	7.17%

(Note: Percentages in each row sum up to the non-partitioned percentage in table II.)

TABLE IV
SCM-4.0 INTRA SUB-MODE STATISTICS (QP = 22)

CU Width	Intra(0) DC	Intra(1) Planar	Intra(10) Horizontal	Intra (26) Vertical	Intra Others
64	36.93%	6.15%	12.43%	42.74%	2.02%
32	16.34%	23.62%	26.85%	22.36%	10.84%
16	18.07%	13.44%	22.73%	16.05%	29.70%
8	24.82%	14.66%	16.71%	15.66%	28.15%

Given the limited paper space, only QP22 simulation data for YUV input are provided, which are summarized into Table II, Table III and Table IV, respectively.

Based on these results, combined with codec implementation and existing literature, we may draw the following conclusions:

1) Table II: larger CUs are more likely to be partitioned. The partition prior probability goes from 86% down to 60% when CU size reduces from 64 to 16.

2) Table III: On CU64 level, PLT mode and IBC-Inter mode are disabled by default. Only IBC-skip and IBC-merge are enabled, but they are chosen rarely. The IBC mode utilization increases as CU size decreases. This is reasonable since smaller blocks have higher likelihood to find perfect or good matches. PLT mode has smaller utilization percentage at all CU levels but larger bit consumption percentage, as studied in [38].

3) Table IV: Unlike natural images, SC image directions are dominated by purely horizontal and purely vertical patterns. The DC, Planar, Horizontal and Vertical sub-modes may cover a large percentage of the Intra mode utilization. For instance, at CU64, these four sub-modes cover almost 98% of intra modes.

4. Visual Analysis and Rate Evaluation: If SC image pattern is horizontal or vertical (e.g.: bicolor CU containing two stripes), the RD selection becomes unpredictable among PLT, IBC and Intra modes. However, the bit consumptions using all these modes are relatively close. Smoothly-varying CUs will be coded using Intra DC or Planar typically without further partitioning. For SC blocks, if a child sub-CU can find a perfect or good match in the previously-coded area, SCM encoder will usually choose to partition. If all sub-CUs in the current block cannot find good matches, SCM-4.0 encoder will preferably choose PLT mode for the whole block without further splitting.

TABLE V
SCM-4.0 MODE COMPLEXITY STATISTICS IN PERCENTAGE

CU Width	Intra	IBC Merge & Skip	IBC Inter	PLT	Total
64	6.93%	3.98%	0.00%	0.00%	10.92%
32	7.72%	6.44%	0.02%	2.34%	16.52%
16	7.44%	8.33%	6.65%	2.06%	24.48%
8	20.83%	4.87%	20.27%	2.11%	48.08%

C. SCM-4.0 Complexity Distribution

We also conduct a survey on SCM-4.0 encoder complexity distribution, using CPU tick counter to document the clock cycles consumed by target encoding mode. Though this complexity profiling result may differ from platform to platform, we assume the percentage of each mode will not change significantly and should reflect the encoder complexity distribution.

The SCM encoding process can be roughly defined as two workflows: One is used for encoder mode selection, later referred as the "Search Process." The other is used for actual binarization into bitstream, later referred as the "Encoding Process." Given the space limit, here we will only present the profiling results for "Search Process" since this workflow consumes over 85% of encoder complexity and is typically the "bottleneck" of SCM encoding. The profiling is conducted on each CU size, over different encoding modes. The distribution is provided in Table V. Please note this result reflects the averaged complexity across sequences. Per-sequence result may vary slightly, depending on the video contents.

From Table V, we may draw the following conclusions:

1) The results coincide with SCM-4.0 codec implementation. The IBC complexity at CU64 and CU32 levels are negligible since IBC-Inter mode is disabled at CU64 level and IBC-Inter mode only checks 64 previous BVs at CU32 level. IBC AMVP is proved very efficient at higher CU levels. At CU16 level, IBC complexity increases due to the additional 1D search within horizontal / vertical line. At CU8 level, IBC complexity increases again significantly due to the global hash-based search and PU calculation.

2) The major complexity is consumed by Intra and IBC-Inter modes. The CU8 Intra mode and IBC-Inter mode consume over 40% of total "Encoder Search" complexity. A simple yet accurate "Natural Image Block (NIB)—Screen Content Block (SCB)" classifier at this CU level may significantly reduce the encoder complexity, by testing only Intra mode for NIBs or SCC modes for SCBs.

IV. ML-FSCC ENCODER DESIGN

In this section, we present our proposed encoder framework with three pre-designed decision tree classifiers.

A. Encoder Workflow Introduction

As illustrated in Fig. 4, the proposed encoder includes three major classification processes: In the first process, input

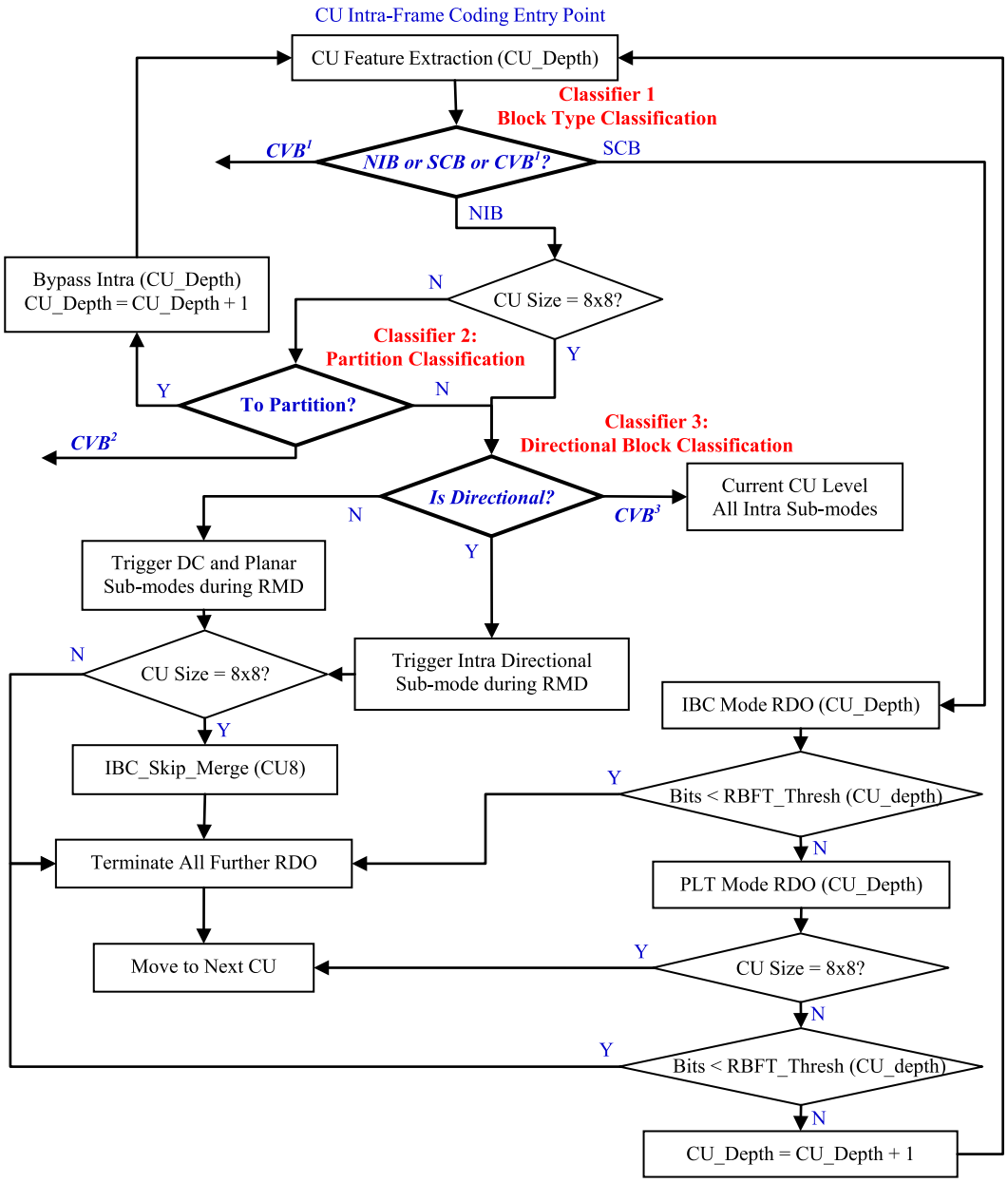

Fig. 4. Proposed ML-FSCC encoding workflowCVB^1 will go through both *NIB* and *SCB* branches; CVB^2 will go through both *Partition* and *Non-Partition* branches. Details are summarized in Table VI.

blocks are classified into either an NIB or an SCB. For NIBs, only Intra modes will be considered at the current CU level, while for SCBs, only SCC modes (i.e., IBC and PLT) will be checked at the current CU level. In the second process, NIBs are classified into either "Partitioned Blocks" (*P-Block* s) or "Non-Partitioned Blocks" (*NP-Block* s). *P-Block*s will bypass current level Intra mode and enter the next-level CU processing directly. For NP-Blocks, the encoder will only check current level Intra modes and immediately terminate further splitting. In the last process, NP-Blocks are classified as "Directional Blocks" (*D-Block*s) or "Non-Directional Blocks" (*ND-Block*s). For *ND-Block* s, the encoder checks both *Intra-DC* mode and *Intra-Planar* mode. For *D-Block* s, the encoder determines the dominant edge direction (as explained in Section V-F) and then triggers the corresponding Intra directional sub-mode. For simplicity, our classifiers make decisions based on the features computed inside the current CU. Thus, it is impossible for the classifier to predict whether a block has a closely-matched block in the previously-coded area using these features. Therefore, the block type classifier may erroneously classify a "Natural Image Look-alike Block" as an *NIB*, when this block happens to match a previously-coded region and is coded more efficiently using an *IBC* mode. Because such situation mainly happens at *CU8* level, therefore, even for a *CU8* classified as a *NIB*, we still check the *IBC-Skip* mode, because this mode only checks a small number of candidates and can often find the best match without going through full IBC search.

TABLE VI
CLASSIFIER PARAMETER SETTINGS COMPARISON AND CVB HANDLINGS

Encoding Mode Setting	*Rate-Distortion Preserving (RDP)*					*Complexity-Reduction Boosting (CRB)*			
Block Type Classification Confidence Threshold	90%					80%			
NIB Partition Classification Confidence Threshold	85%					80%			
NIB Directionality Classification Confidence Threshold	*85%*					80%			
Rate-based Fast Termination (RBFT) Thresholds	*10 percentile at encoding bits CDF*					*50 percentile at encoding bits CDF*			
	QP	*22*	*27*	*32*	*37*	*22*	*27*	*32*	*37*
	CU32	*≤8*	*≤8*	*≤8*	*≤8*	*≤72*	*≤48*	*≤32*	*≤24*
	CU16	*≤8*	*≤8*	*≤8*	*≤8*	*≤32*	*≤24*	*≤24*	*≤16*
	CU8	*≤8*	*≤8*	*≤8*	*≤8*	*≤56*	*≤40*	*≤32*	*≤24*
CVB[1] Handling Strategy (check NIB, SCB and Partition branches)	*SCB*	*SCC Full-RD at the current CU with RBFT*							
	NIB	Intra Full-RD at current CU				Intra Fast Decision *following NIB processing branch*			
	Partition	Check *RBFT*. If not terminated, proceed with next CU level							
CVB[2] Handling Strategy (check both *Partition* and *Non-Partition* branches)	*Non-Partition*	Intra Full-RD at current CU				Intra Fast Decision *following NP-Block processing branch*			
	Partition	Check *RBFT*. If not terminated, proceed with next CU level							
CVB[3] Handling	All Intra Sub-modes at the current CU level								

For each classification, we use a "soft-decision" classifier that outputs a decision confidence level. When the decision confidence for a CU is below a preset threshold, this CU is defined as a "Controversial Block" (*CVB*). Throughout this paper, the *CVB*s in the first classification process (i.e., " *NIB* versus *SCB* Classification") are denoted as CVB^1. The *CVB*s in the second classification process (i.e., "Partition versus Non-Partition Classification") are denoted as CVB^2. The *CVB*s in the third classification process (i.e., "Directional versus Non-Directional Classification") are denoted as CVB^3. We propose two encoding settings with different *CVB* handling strategies. Under " *Rate-Distortion Preserving* " (*RDP*) setting, for CVB^1s, the encoder will evaluate both Intra and SCC modes at the current level and then examine the RD performance with CU partition, where each sub-CU will be processed according to the same workflow in Fig. 4. For CVB^2s, the encoder will evaluate all the Intra modes for the current CU level and then examine the RD cost with CU partition. Under " *Complexity Reduction Boosting* " (*CRB*) setting, rather than going through Intra mode full-RD evaluations, we allow fast Intra partition and sub-mode decisions: For CVB^1s, the encoder will firstly follow the *NIB* branch in Fig. 4 to evaluate Intra modes and then check SCC modes. For CVB^2s, the encoder will evaluate Intra mode at the current level with fast directional block classification invoked and then proceed with CU partition and sub-CU processing. For CVB^3s, the encoder will evaluate all the Intra sub-modes for both *RDP* and *CRB* settings. The strategies are summarized in *Table VI*. The confidence thresholds can be chosen according to the desired trade-off between encoder RD-performance and complexity saving. A higher confidence threshold will preserve RD better while a lower confidence threshold will reduce the complexity more. We allow different threshold settings at different CU levels, because higher level CU misclassifications may affect RD performance more than lower level CU misclassifications. We describe the classifiers for each process and how we derive these classification confidence levels in details in Sec. V.

At CU64 level, by default, SCM-4.0 already disabled PLT and IBC-Inter modes. IBC Merge and Skip modes utilization prior probability is extremely low (2.55% as in Table III). By disabling Merge/Skip at CU64 level, a 4.5% complexity reduction is obtained with negligible BD-loss, since the small amount of CU64 Merge or Skip blocks could still be efficiently coded by CU32 level Merge/Skip mode. Based on such statistics, at CU64 level, only Intra mode is enabled and block type classifier is not designed at this level.

B. Rate-based Fast Termination (RBFT)

When SCM finished the RD calculation for a specific mode, the optimal RD cost will be updated and the corresponding best mode will be documented if the current mode outperforms the previous best coding mode. In addition to reducing the mode candidates using the three classifiers, we further incorporate a rate-based fast CU termination process in a similar formulation as described in [39]. To summarize, when the current mode bit consumption is lower than our statistics-based threshold at the corresponding CU level, we assume the current coding mode is sufficiently efficient and the remaining modes are terminated. *RBFT* is invoked for *SCB*, CVB^1 and CVB^2.

For an SCB, we successively check IBC-Skip, IBC-Merge, IBC-Inter and PLT modes at the current CU level. For a CVB^1, we check Intra, IBC-Skip, IBC-Merge, IBC-Inter, and PLT modes at the current CU level. Whenever the rate required for the target mode is below the threshold, *RBFT* will be invoked and the following RDO processing (including CU partitioning) will be terminated immediately. For a CVB^2, we only check Intra coding at the current CU level. If the Intra coding rate is below the RBFT threshold, CU partitioning will be terminated.

To derive the rate thresholds, we determine the Cumulative Distribution Function (*CDF*) of the encoding bits consumed at each CU level using our ground-truth data. We choose the thresholds based on a target percentile of the *CDF*. The *RBFT* parameters are summarized in Table VI. For simplicity, we approximate and quantize the collected coding bits into bytes (i.e., multiple of eight bits) in our implementation.

C. Coding Efficiency and Coding Complexity Trade-off

The proposed framework allows us to adjust the classifier confidence thresholds and *RBFT* thresholds together to balance the coding efficiency and complexity. For example, when the confidence threshold is increased, more blocks will be defined as "controversial" and will go through full-RD mode checking, and therefore better preserve coding performance. When the confidence threshold is reduced, more blocks will be directly classified and bypass unnecessary coding modes checking, leading to more complexity saving. Similarly, a larger *RBFT* threshold setting will terminate more CU mode checking and CU splitting and therefore promote encoding speed while a smaller *RBFT* threshold will preserve RD performance better.

V. Feature Selection and Classifier Design

In this section, we describe the features used to drive these classifiers and the criteria used to select the classifier type, the classifier training procedures, and the final trained classifier structures.

A. Feature Selection

Statistically we have some prior knowledge on SC videos and the relationships between SC image patterns and mode selection. For instance, "Homogenous" regions are more likely to be encoded in larger CUs with Intra mode. PLT-coded CUs are more likely to contain limited distinct color numbers and sharp edges. Discontinuous-tone SC areas are more likely to be encoded using IBC or PLT modes than Intra mode. Larger CUs are more likely to be further partitioned than smaller CUs, especially under low QP settings, and so on.

With such prior knowledge, we directly derive features for our fast mode and partition decision tasks and then apply the supervised-learning approach, rather than learning the features from the raw image blocks using deep learning techniques. The features we used for different classifiers are summarized as follows.

Feature 1: Sub-CU Horizontal and Vertical DC Difference (HVDD) formulated in (1)–(3):

$$HDD = |\mathrm{DC}_1 - \mathrm{DC}_2| + |\mathrm{DC}_3 - \mathrm{DC}_4| \tag{1}$$

$$VDD = |\mathrm{DC}_1 - \mathrm{DC}_3| + |\mathrm{DC}_2 - \mathrm{DC}_4| \tag{2}$$

$$HVDD = min\{HDD, VDD\} \tag{3}$$

where HDD and VDD are intermediate horizontal and vertical measures of sub-CU DC value difference between horizontally and vertically adjacent sub-CUs. "|.|" is the absolute value operator. The sub-index indicates the corresponding sub-CU location. For instance, DC_1 corresponds to the DC value of first sub-CU located at the upper-left corner of the current CU. A CU with a smaller HVDD value has a stronger horizontal or vertical directionality.

Feature 2: CU Variance as defined in (4), where $Y(x, y)$ is luminance value at pixel location (x, y) and $\bar{Y}$ is the average luminance value over the current CU and N is the CU width. Variance is a good indicator of block smoothness. A CU with smaller variance is less likely to be further partitioned

$$\sigma^2 = \frac{1}{N^2} \sum\nolimits_{(x,y)\in CU} \left(Y\left(x, y\right) - \bar{Y}\right)^2. \tag{4}$$

Feature 3: CU Gradient Kurtosis (GK). In order to evaluate whether a block has a dominant direction, we compute the histogram of the gradient orientation and then compute the orientation histogram kurtosis, which measures the histogram peakiness. To calculate this feature, the CU gradient maps are firstly derived by convolving the input image with "Sobel" masks. Let $Y_H(x, y)$ and $Y_V(x, y)$ denote the horizontal and vertical gradient at pixel location (x, y) and $Mag(x, y)$ and $Ang(x, y)$ denote the magnitude and orientation of the gradient computed based on $Y_H(x, y)$ and $Y_V(x, y)$. We firstly apply a threshold (with value 30 for *CU64* and *CU32* and 10 for *CU16* and *CU8*) on gradient magnitude map to filter out small local texture variation. From these thresholded gradient maps, we compute the gradient angle histogram $G(\theta)$, which denotes the sum of gradient magnitudes of all pixels for each gradient angle θ between 0 and 180 ° (exclusive) with a step of 1 °. Finally, Gk is defined in (5), where $\bar{G}$ is the averaged gradient magnitude over the whole gradient direction histogram

$$GK = N^2 \frac{\sum_{\theta\in[0,180)} \left[G(\theta) - \bar{G}\right]^4}{\left(\sum_{\theta\in[0,180)} \left[G\left(\theta\right) - \bar{G}\right]^2\right)^2}. \tag{5}$$

Feature 4: CU Gradient Magnitude Peak (GMP), which is the gradient magnitude that achieves the peak over the gradient magnitude histogram (excluding zero-gradient). GMP indicates the most frequent nonzero gradient magnitude in a CU. SCBs usually consist of sharper edges and thus have larger GMP values. The reason that we exclude zero-gradient is because such gradient is most likely to be the peak value in the majority of the blocks and does not facilitate the differentiation between NIBs and SCBs.

Feature 5: Zero Gradient Percentage (ZGP), defined as the ratio between the pixel number with zero gradient magnitude and the CU area. SCBs mostly contain a large area of constant background color and therefore have larger ZGP values than smoothly-varying NIBs.

Feature 6: CU Color Number (CN). To calculate this feature, RGB or YUV components are firstly combined into a 24-bit "color triplet". The number of distinct "color triplet" inside the current CU is counted to deriveCN.

B. Classifier Selection

Recently, several groups have considered the application of machine learning techniques for fast mode decision in video coding and transcoding ([17], [34], [40]–[43]). In our framework, several popular supervised learning methods are evaluated, including "Supported Vector Machine" (SVM), "Neural Networks" (NN), and "Decision Tree" (DT). All these three classifiers provide similar accuracy for our classification problem when using the same feature set. For SVM, because our data samples are not linearly separable, Gaussian Radial Basis Function Kernel (RBF) is used to implicitly map input features onto high-dimensional space. Thus, all the supporting

vectors (SVs) derived during training stage have to be stored within the encoder memory for future prediction and the number of SVs tends to grow larger when training samples are increased. For instance, the number of SVs is over 1600 on average when we include 10 000 random training samples. This will not only consume valuable encoder on-chip memory but also make the resultant classifier less adapted to new CU data. For the NN, we used a single hidden layer structure and optimized the number of hidden nodes and other network parameters through a validation process. Both NN and DT involve minimal processing complexity and can be incorporated in the encoder easily. However, the derived NN classifier structure (including weighting factors, bias terms, etc.) does not provide a good interpretation. On the other hand, DT structure could be easily implemented as a set of "if-then" rules. These derived conditions typically coincide with human observations and analytical reasoning, which provide valuable insights about the features and what ranges of the features are typically associated for different classes. Taken into account the prediction accuracy, model simplicity and interpretability and memory consumption, DT is adopted as our learning model. Details about the DT classifier training are provided in Sec. V.C.

C. Decision Tree Parameter Training

All the classifiers are trained separately for each CU size. Our training data initially contains totally 191 200 CU64 samples, 764 800 CU32 samples, 3 059 200 CU16 samples and 12 236 800 CU8 samples from the sample frames in Table I for QP22, QP27, QP32, and QP37, respectively. If the higher CU level ground-truth decision is "Not to Partition," all its sub-CU data samples will be deactivated and removed from our training set. Since we have sufficient samples, cross-validations are assumed unnecessary and thus not used. For simplicity, on each CU level, we randomly select half samples to form the Training Set (TRS) and the other half samples to form the Validation Set (VLS). TRS is used to derive the DT model and parameters (e.g., the decision variables and the decision thresholds at each tree node) and VLS is used to evaluate the generalization error and prune the derived trees. Experiments show that the division setting between TRS and VLS will not change VLS prediction accuracy significantly since we have a large number of training and validation samples.

The ground-truth mode and partition decision labels for CU samples are obtained by encoding the sample frames in Table I using SCM-4.0 encoder with AI encoding structure and default settings in CTC [37]. For each CU sample, if the entire CU (including its sub-CUs, if partitioned) is encoded using an Intra mode, the block is labeled as an "NIB." If the entire CU is encoded using an SCC mode (IBC or PLT), or if at least one sub-CU is coded using an SCC mode, the CU is labeled as an "SCB." The training is implemented using MATLAB built-in "Classification and Regression Tree" (CART) module in "Statistics Toolbox" (Ver. 8.3) [44]. We also compared the prediction accuracies with other decision tree implementations (e.g., DT-C5.0 in [45]) and the performances are similar.

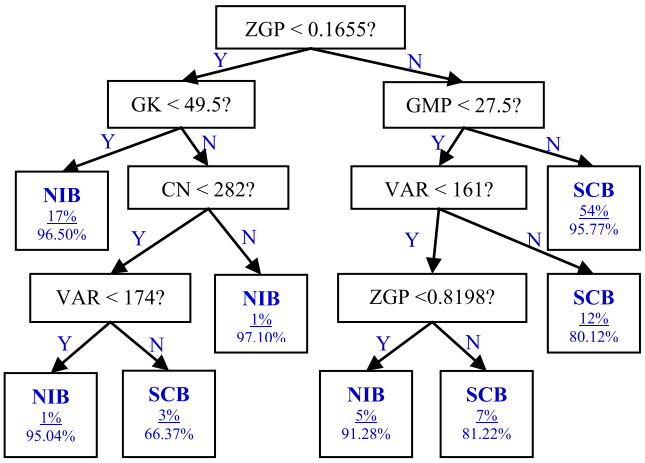

Fig. 5. CU32 NIB-SCB classification decision tree.

The feature priority is determined based on maximum deviance reduction (or equivalently, the cross entropy). Namely, the feature that provides larger deviance reduction is given a higher priority and appears earlier in the decision tree. We use the sample number in the DT node as the stopping criterion and stop splitting a node as soon as the total number of training samples in the node is less than or equal to 1% of the TRS size. Then the tree structure is progressively pruned node by node backward while we track the VLS prediction accuracy until the maxima is acquired. The detailed decision tree structures for each classification tasks are provided in the following Sections.

D. Block Type Classification: NIB Versus SCB

In this process, we train a decision tree using all the features to classify the incoming CUs as either an NIB or an SCB. Training blocks for NIBs include blocks coded fully using Intra-mode. Training blocks for SCBs include blocks coded using IBC and PLT modes only, or a mixture of Intra and SCC modes. The derived decision trees for different CU levels after pruning are provided in Figs. 5–7. Please note that in our design, at CU64 level, only Intra mode is enabled, as described previously in Section IV-A, so no block type classifier is designed at this level.

For each decision tree leaf node, we show two numbers (in percentages) obtained during the training stage. The top number (i.e., the node probability) indicates the percentage of samples going to this node among all the training samples. The bottom number (i.e., the node accuracy) reveals the percentage of samples correctly classified amoung all training samples in that node. A low node accuracy indicates that the samples landing in that node cannot be classified with high confidence. In our work, we consider the node accuracy as the decision confidence. Any block landing in a leaf node with confidence lower than a preset threshold in this process will be defined as a CVB^1. Recall that CVB^1 will go through both the SCB and NIB branches for the mode and partition decisions.

E. Partition Decision: Partition Versus Non-Partition

In this process, we train another decision tree to classify NIBs into "Partitioned Blocks" (P-Blocks) or "Non-Partitioned

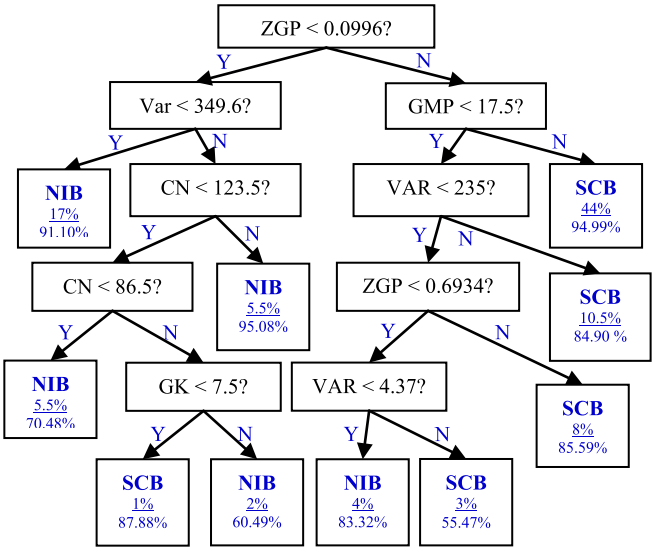

Fig. 6. CU16 NIB-SCB classification decision tree.

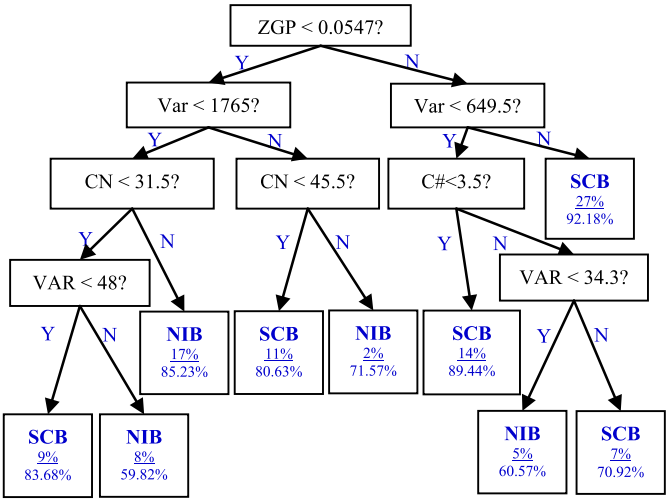

Fig. 7. CU8 NIB-SCB classification decision tree.

Blocks" (NP-Blocks) based on CU homogeneity. Statistically, NIBs with larger variance, sub-CU mismatch and smaller gradient kurtosis are more likely to be further partitioned. Therefore, we use features {1, 2, 3} to train this classifier for each CU level. HVDD and CU Variance describe Sub-CU homogeneity, while GK reflects CU orientation homogeneity. P-Blocks with high confidence level will directly bypass current level Intra mode selection and directly enter next-level CU processing. NP-Blocks with high confidence level will only examine current level Intra mode and immediately terminate further splitting. The trained decision trees for different CU levels are provided in Figs. 8–10.

F. Sub-Mode Decision: Directional vs. Non-Directional

To classify NP-Blocks into "Directional Blocks" (D-Blocks) or "Non-Directional Blocks" (ND-Blocks), we use features {1, 2, 3} because directional CU usually has a dominant direction that can be reflected by high GK value,

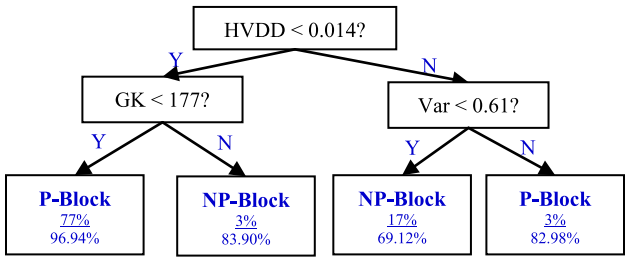

Fig. 8. CU64 NIB NP/P-block classification decision tree.

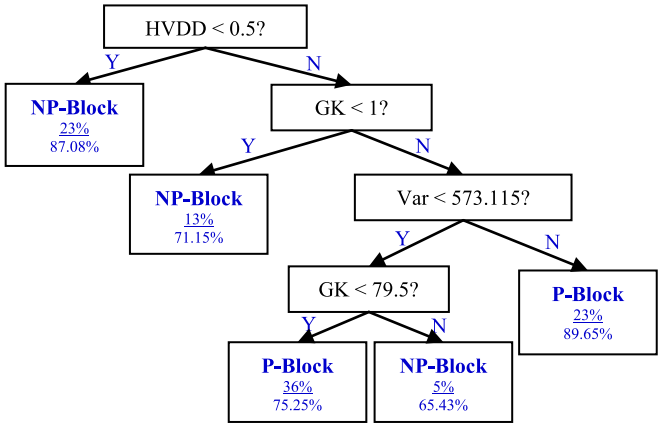

Fig. 9. CU32 NIB NP/P-block classification decision tree.

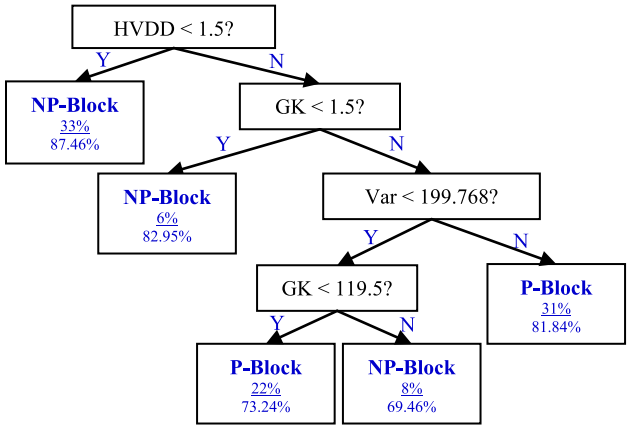

Fig. 10. CU16 NIB NP/P-block classification decision tree

while non-directional CU is usually smooth with small HVDD, Variance and GK. The resultant classifiers are provided in Figs. 11–14.

ND-Blocks include Intra DC and Planar mode. D-Blocks include 33 directional Intra sub-modes. If an NIB is classified as a D-Block with a high confidence level, we firstly determine the Dominant Edge Direction (DED), which is the gradient angle corresponding to the highest peak in the gradient angle histogram, quantized into an integer degree between 0 and 179. Then we map this direction angle into an Intra directional sub-mode index (between 2 and 34), according to (6) and (7), where "deg" is the degree integer value between 0 and 179 and $\lfloor . \rfloor$ is the rounding operation to the next smaller integer.

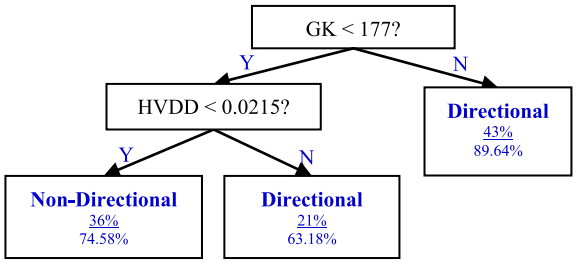

Fig. 11. CU64 directional/non-directional classification decision tree.

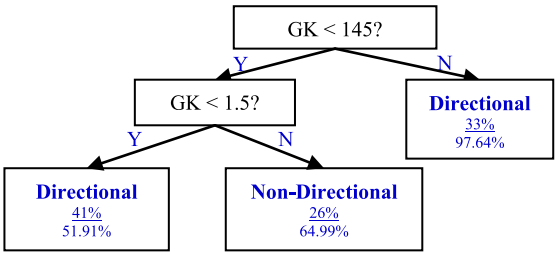

Fig. 12. CU32 directional/non-directional classification decision tree.

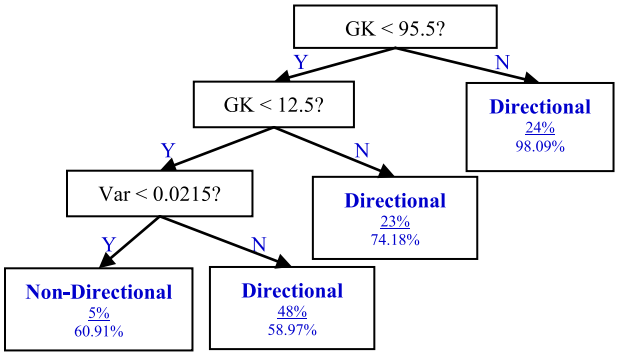

Fig. 13. CU16 directional/non-directional classification decision tree.

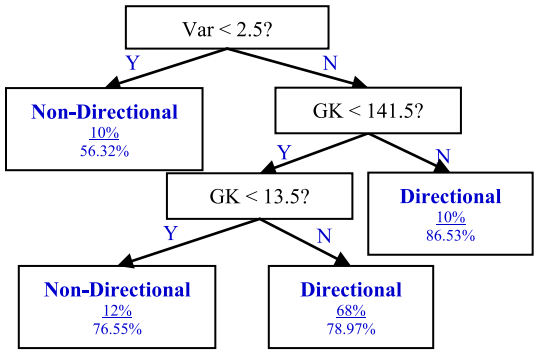

Fig. 14. CU8 directional/non-directional classification decision tree.

Later we only use this mapped sub-mode for Intra coding.

$$Mode = \lfloor \deg/5.5 \rfloor + 10 \quad deg \in [0\ 135] \tag{6}$$

$$Mode = 10 - \lfloor (180 - \deg)/5.5 \rfloor \quad deg \in (136\ 179]. \tag{7}$$

G. Exploration of Other Classifiers for Fast Decision

In addition to the three classification processes, we also investigated to design a classifier that can make fast partition decisions for SCBs. Specifically, We attempted to design a classifier to separate "Pure SCBs" and "Mixed SCBs", where "Pure SCBs" refer to those blocks fully encoded using SCC modes at the current CU or the sub-CUs and "Mixed SCBs" refer to those blocks that are partitioned and coded using both Intra and SCC modes at the sub-CU levels. With such a classifier, for "Mixed SCBs," we can directly bypass the mode decision at the current CU level. However, the classifier we designed cannot provide sufficient accuracy for the majority of the SCBs. This is because SCB partition decisions are primarily determined by IBC search rather than local statistical features, and it is infeasible to accurately determine whether a block should be partitioned only using the features collected within the current CU. Given "Mixed SCBs" only consume a small percentage (For instance, in "WebBrowsing" sequence, there are only 2.83% "Mixed SCBs" at CU16 level; in "Console" sequence, there are only 2.79% "Mixed SCBs" at CU16 level.), we decide to disable this classification stage in our framework.

VI. Experimental Results and Evaluations

Our proposed ML-FSCC Intra-frame coding framework is evaluated and compared with SCM-4.0 anchor software, strictly following the CTC defined in [37]. 13 standard SCC sequences are evaluated under 4 QP settings (22, 27, 32, and 37) under All-Intra (AI) configurations. To verify that our proposed classifiers are generalizable for new SC videos, we further evaluate our proposed framework over unseen SC testing sequences (from [46], [47], [48] and [49]) previously proposed in JCTVC community. The coding performances are evaluated using homogeneous Windows 7 (64-bit) desktops with Intel-i5 CPU (2.67 GHz dual cores) and 4GB RAM.

The coding performance is measured using BD-Rate [50] against SCM-4.0 encoder. The complexity saving is measured directly using the relative reduction of the encoding time, as defined in (8), where T_{Anchor} is the encoding time of SCM-4.0 encoder and $T_{Proposed}$ is the encoding time of our proposed encoder schemes for the same sequence

$$\Delta Comp = \frac{T_{Anchor} - T_{Proposed}}{T_{Anchor}} \times 100\%. \tag{8}$$

Two sets of simulation results are provided with different encoding settings. The "Rate-Distortion Preserving" (*RDP*) setting focuses more on coding efficiency protection, while the "Complexity Reduction Boosting" (*CRB*) setting focuses on encoder acceleration. The detailed parameters for each setting are provided in Table VI.

Compared with anchor *SCM-4.0* software with the default Full-Frame IBC (FF-IBC) configuration, our proposed fast Intra-coding framework can achieve an averaged complexity reduction of 40% under *RDP* setting with 1.46% negligible BD-loss. Under *CRB* setting, we achieve 52% average complexity reduction with 3.65% acceptable BD-loss. In Table VII, the detailed simulations are provided. Given the space limit, only YUV-444 results are provided. However, the proposed framework and methodologies could be generalized onto RGB4:4:4 sequences and YUV4:2:0 sampling format.

TABLE VII

FULL-SEQUENCE CODING EFFICIENCY AND COMPLEXITY REDUCTION FOR PROPOSED ML-FSCC SCHEME (RDP AND CRB SETTINGS)

Index	Sequence	QP	*Anchor SCM-4.0*			*Rate-Distortion Preserving (RDP)*					*Complexity-Reduction Boosting (CRB)*				
			Rate	*PSNR*	*Time*	*Rate*	*PSNR*	*Time*	△*R*	△*T*	*Rate*	*PSNR*	*Time*	△*R*	△*T*
1	FlyingGraphics* 1080p, YUV444, 300 frames Text & Graphics	22	63151.64	50.81	32953	63542.42	50.81	18908	+1.02%	-40.40%	65906.29	50.74	17386	+4.84%	-46.33%
		27	48315.12	46.29	30212	48730.39	46.30	17781			50400.52	46.25	16289		
		32	36063.76	42.03	27080	36535.04	42.06	16561			37722.30	41.99	14648		
		37	25677.31	37.73	23901	26284.74	37.78	14579			26982.57	37.68	12886		
2	Desktop* 1080p, YUV444, 600 frames Text & Graphics	22	34238.06	55.69	57319	34945.75	55.76	29939	+1.93%	-46.15%	35611.59	55.59	27381	+3.67%	-50.63%
		27	30058.46	50.69	54387	30647.43	50.73	29216			30984.34	50.64	26589		
		32	25902.95	45.32	50997	26503.70	45.44	27635			26829.05	45.26	25319		
		37	21195.32	39.93	47308	21700.44	39.98	26143			22010.24	39.87	24217		
3	Console* 1080p, YUV444, 600 frames Text & Graphics	22	31312.03	58.02	45685	31836.22	58.01	23644	+1.78%	-46.53%	32646.07	57.94	21820	+4.32%	-51.01%
		27	27023.39	53.38	43271	27501.74	53.41	22840			27908.09	53.33	21357		
		32	23010.08	48.80	40385	23499.20	48.94	21748			23863.74	48.80	20034		
		37	18078.58	43.22	35567	18697.40	43.37	19743			19494.02	43.19	17518		
4	WebBrowsing* 720p, YUV444, 300 frames Text & Graphics	22	6687.96	55.60	10952	6771.89	55.65	5509	+1.51%	-48.05%	6902.42	55.55	4630	+3.93%	-57.13%
		27	5246.96	50.77	10096	5335.99	50.82	5197			5419.32	50.71	4310		
		32	3977.55	45.02	9105	4048.90	45.06	4816			4142.35	45.02	3937		
		37	2874.49	39.73	8179	2927.24	39.72	4347			3009.91	39.73	3540		
5	Map* 720p, YUV444, 600 frames Text & Graphics	22	40313.58	51.46	25650	40541.57	51.42	16636	+1.21%	-34.73%	41093.27	51.35	14396	+4.15%	-46.86%
		27	27919.67	47.55	22710	28219.24	47.57	15141			28568.79	47.41	12778		
		32	18037.49	43.73	19950	18364.30	43.75	13198			18512.19	43.54	10312		
		37	11336.96	40.06	17257	11638.47	40.14	10937			11657.84	39.98	8366		
6	Programming* 720p, YUV444, 600 frames Text & Graphics	22	37599.61	48.68	24191	38045.05	48.63	15431	+1.33%	-36.46%	39183.13	48.63	12902	+5.09%	-47.50%
		27	25865.49	44.77	21950	26074.49	44.71	14086			27056.16	44.72	11720		
		32	17706.54	41.12	19882	17760.62	41.04	12687			18530.46	41.08	10334		
		37	11921.38	37.62	18003	11988.65	37.57	11234			12544.04	37.61	9236		
7	SlideShow* 720p, YUV444, 500 frames Text & Graphics	22	4384.21	54.81	11434	4445.39	54.78	5693	+2.52%	-52.99%	4473.51	54.69	4694	+ 3.82%	-62.01%
		27	2763.88	51.90	10421	2818.10	51.87	5060			2810.52	51.79	4055		
		32	1753.24	48.92	9576	1777.59	48.91	4447			1797.47	48.89	3528		
		37	1096.42	45.89	8885	1161.58	45.86	3844			1159.82	45.79	3125		
8	BasketballScreen* 1440p, YUV444, 300 frames Mixed-Content	22	120053.41	48.96	47808	121076.25	48.94	31794	+1.36%	-33.53%	124520.02	48.93	24412	+ 4.18%	-49.27%
		27	80648.92	45.68	43509	81395.48	45.66	28847			83551.58	45.64	22120		
		32	55455.76	42.46	38235	56089.96	42.42	25551			57210.17	42.37	19293		
		37	37503.60	38.99	33535	38080.86	39.00	22216			38733.71	38.90	16956		
9	MissionControlClip2* 1440p, YUV444, 300 frames Mixed-Content	22	98901.72	49.75	44730	102348.96	49.74	25491	+2.57%	-42.86%	102406.72	49.69	19701	+ 4.46%	-57.50%
		27	64867.49	46.78	37717	66545.72	46.75	22288			67179.25	46.72	16922		
		32	40736.22	43.70	32218	41199.29	43.65	18681			42071.64	43.64	13526		
		37	24628.60	40.54	28430	25002.68	40.50	15491			25511.05	40.45	11123		
10	MissionControlClip3* 1080p, YUV444, 300 frames Mixed-Content	22	66768.84	49.31	54282	67915.02	49.29	33477	+1.80%	-38.55%	69056.75	49.24	26190	+ 4.29%	-52.30%
		27	45863.04	45.72	48522	46626.64	45.71	30076			47489.26	45.69	23449		
		32	31204.37	42.32	42513	31662.35	42.29	26311			32391.01	42.26	20244		
		37	20477.63	38.72	37329	20824.04	38.72	22491			21358.51	38.67	17395		
11	Robot* 720p, YUV444, 300 frames Animation	22	37773.46	44.50	16220	37846.70	44.46	11304	+1.19%	-32.02%	38222.42	44.45	7942	+ 3.26%	-55.39%
		27	18192.87	41.27	13549	18233.27	41.24	9452			18512.26	41.23	6297		
		32	8009.24	38.70	10949	8044.67	38.68	7540			8236.34	38.67	4720		
		37	3756.62	36.65	8431	3787.28	36.61	5363			3864.10	36.60	3364		
12	EBURainFruits+ 1080p, YUV444, 250 frames Camera-captured	22	88190.35	46.36	25396	88602.90	46.36	17949	+1.19%	-30.15%	88703.43	46.36	12468	+ 1.54%	-52.20%
		27	52217.65	44.14	21666	52513.63	44.14	15540			52600.31	44.13	10597		
		32	29742.03	41.80	18354	30090.67	41.79	13002			30149.20	41.78	8601		
		37	16334.27	39.42	15504	16646.76	39.38	10260			16670.64	39.38	7182		
13	Kimono1+ 1080p, YUV444, 120 frames Camera-captured	22	45867.85	42.15	12579	46106.57	42.14	8741	+1.46%	-31.62%	46159.07	42.15	6750	+ 1.38%	-49.45%
		27	14129.30	40.53	9533	14268.78	40.52	6734			14272.65	40.52	4893		
		32	7487.77	39.37	7543	7574.23	39.37	5278			7578.20	39.37	3647		
		37	4017.99	37.99	5933	4075.33	37.98	3762			4076.24	37.98	2900		
14	Doc+ 720p, YUV444, 500 frames Text & Graphics	22	2858.43	54.00	15290	2886.31	54.00	9009	+1.34%	-41.69%	2935.62	53.89	7149	+ 3.59%	-53.32%
		27	2290.39	49.99	14402	2311.75	49.97	8348			2346.67	49.90	6774		
		32	1799.81	45.70	13219	1817.94	45.60	7800			1853.31	45.53	6182		
		37	1338.44	41.06	11931	1357.08	41.02	6843			1382.78	40.93	5506		
15	PptDocXls+ 720p, YUV444, 200 frames Text & Graphics	22	11488.54	56.62	13335	11682.88	56.73	7706	+1.54%	-39.95%	11866.87	56.63	6697	+ 2.66%	-47.76%
		27	10144.84	52.19	12579	10290.04	52.16	7446			10405.03	52.04	6573		
		32	8886.82	47.49	11759	9018.64	47.47	7168			9109.02	47.45	6218		
		37	7422.76	41.44	10523	7555.90	41.47	6552			7710.86	42.24	5641		
16	TwistTunnel+ 720p, YUV444, 300 frames Text & Graphics	22	8856.53	60.52	11641	8929.94	60.62	7645	+0.45%	-33.92%	9385.69	60.04	6211	+ 4.27%	-44.24%
		27	7817.87	56.76	11449	7870.86	56.92	7560			8148.29	56.62	6432		
		32	6953.51	52.37	10902	7009.09	52.43	7227			7223.70	52.41	6238		
		37	5948.58	47.05	10385	6013.24	47.26	6887			6139.40	47.18	5845		
17	VideoConferencingDocSharing+ 720p, YUV444, 300 frames Text & Graphics	22	8649.28	53.37	10350	8769.16	53.36	5876	+1.66%	-42.63%	8918.43	53.32	4831	+ 3.69%	-52.84%
		27	6849.65	49.50	9718	6951.42	49.45	5559			7057.88	49.40	4579		
		32	5598.53	44.98	8979	5692.60	45.00	5189			5796.81	44.90	4288		
		37	4476.89	40.52	8606	4565.34	40.55	4966			4661.47	40.63	4052		
18	Viking+ 720p, YUV444, 300 frames Animation	22	24748.31	47.11	13974	24803.89	47.09	10111	+1.42%	-29.02%	24923.48	47.06	6815	+ 3.59%	-55.30%
		27	13455.97	44.49	12027	13504.16	44.46	8758			13589.60	44.41	5612		
		32	6867.27	41.96	9872	6902.66	41.92	7098			6949.05	41.85	4257		
		37	3369.09	39.46	7849	3395.93	39.40	5246			3434.82	39.35	3159		
19	Web+ 720p, YUV444, 300 frames Text & Graphics	22	7871.61	57.47	19958	7934.66	57.42	9857	+0.91%	-49.64%	8058.97	57.27	9466	+ 2.47%	-51.79%
		27	6955.36	52.29	19136	7014.49	52.28	9583			7114.02	52.22	9195		
		32	5882.23	46.36	18111	5929.76	46.38	9151			6016.90	46.32	8754		
		37	4569.58	40.30	16632	4606.26	40.15	8556			4666.41	40.09	8157		
20	WordEditing+ 720p, YUV444, 300 frames Text & Graphics	22	23352.03	50.59	27511	23540.64	50.59	15416	+1.09%	-43.87%	24032.26	50.52	13735	+ 3.71%	-49.77%
		27	17180.68	46.28	25391	17359.40	46.27	14441			17699.56	46.21	12897		
		32	12619.35	42.35	23736	12754.04	42.34	13146			13049.04	42.28	11731		
		37	9078.95	38.60	20517	9174.08	38.59	11537			9385.32	38.57	10419		

(***Rate***: in kbps; ***PSNR***: Y/U/V average PSNR in dB; ***Time***: Encoding time in Second. ***△R***: BD-Rate Increment in Percentage. ***△T***: Encoding Time Reduction in Percentage)
(Sequences with * are those videos whose 10 frames were used for training. Results reported are whole sequence encoding statistics according to CTC.
Sequence with + are not used during training and reported to validate machine learning model generalization.)

TABLE VIII

60-FRAME CODING EFFICIENCY AND COMPLEXITY EVALUATIONS FOR PROPOSED CLASSIFIERS (UNDER *RDP* SETTING)

Category	*Text & Graphics (TG)*		*Mixed Contents (MC)*		*Animation (AM)*		*Camera-Captured (CC)*	
Performance vs. SCM-4.0	⊿R	⊿T	⊿R	⊿T	⊿R	⊿T	⊿R	⊿T
SCM-4.0 + Classifier 1	+0.94%	-31%	+0.50%	-36%	+0.26%	-23%	+0.32%	-21%
SCM-4.0 + Classifier 1 + Classifier 2	+1.13%	-34%	+1.00%	-40%	+1.21%	-28%	+1.24%	-27%
SCM-4.0 + Classifier 1 + Classifier 2 + Classifier 3	+1.06%	-35%	+1.14%	-41%	+1.51%	-30%	+1.34%	-29%
SCM-4.0 + Classifier 1 + Classifier 2 + Classifier 3 + RBFT	+1.51%	-43%	+1.75%	-45%	+1.60%	-31%	+1.49%	-30%

(Anchor is SCM-4.0. ***⊿R***: BD-Rate Increment in Percentage. ***⊿T***: Encoding Time Reduction in Percentage.)

TABLE IX

CODING EFFICIENCY AND COMPLEXITY COMPARISONS WITH PREVIOUS WORK

Previous SCC Fast Encoding Work	*Anchor Codebase*	*cross-sequence Complexity Reduction*	*cross-sequence BD-Rate Loss*	*Simulation Sequences*
Kwon and Budagavi [28]	HM-12.0+RExt-4.1	22%~31%	0.5%	HEVC Class F Sequences and SCC Sequences.
Tsang, Chan and Siu [30]	SCM-2.0	7.6%~29.2%	0.08%~0.66%	Sequences 7, 8 from Table VII, "ChinaSpeed", "SlideEditing".
Zhang, Guo and Bai [32]	HM-12.1+RExt-5.1	32%	0.80%	Sequences 4, 6, 7, 15, 20 from Table VII, "Waveform", "PcbLayout".
Zhang and Ma [33]	SCM-3.0	39%	1.04%	Sequence 1 through 13 from Table VII.
Duanmu, Ma, Wang [34]	SCM-3.0	37%	3.05%	Sequence 1 through 13 from Table VII.
Duanmu, Ma, Wang* [34]	SCM-4.0	35%	3.69%	Sequence 1 through 13 from Table VII.
Proposed ML-FSCC (RDP)	SCM-4.0	40%	1.46%	20 Sequences from Table VII.
Proposed ML-FSCC (CRB)	SCM-4.0	52%	3.65%	20 Sequences from Table VII.

(The 7th entry marked with "*" is our previous algorithm in [34] re-implemented on top of SCM-4.0 software codebase)

From the simulation results, we may draw the following conclusions.

1) Our proposed framework has less gain on the sequences coded mostly in smaller CU sizes, because fewer blocks can be fast-terminated. For instance, "Map" and "Robot" sequences are dominated by small CU8x8 blocks and have the minimum complexity reduction among all the sequences.

2) Our machine learning model is generalizable to unseen SC videos. We observe comparable BD-rate increase and similar complexity reduction. Besides, the proposed framework does not degrade the performances over camera-captured natural videos.

3) Our proposed framework has larger complexity savings for sequences coded using higher QP values, because more SCBs can search and find a perfect or good match under higher QP settings through IBC search.

4) Our proposed framework is scalable and controllable. By adjusting classifier confidence thresholds and RBFT thresholds, we may achieve desired trade-off between encoder efficiency and complexity for various applications. Also, through our experiments, we find that confidence threshold setting at larger CUs is more influential for the efficiency-complexity trade-off than at smaller CUs. A conservative setting at larger CUs may better preserve RD and a radical setting at larger CUs may reduce more computational complexity. Variable confidence thresholds settings across CU size may outperform a uniform setting across CU size.

5) RBFT also facilitates the complexity reduction. However, unless the rate-thresholds are set conservatively, the RD-loss may increase significantly for specific sequence, because of the diversity of SC image patterns. A possible solution to improve RBFT process is through online adaptation of rate-thresholds based on the bit-rate distributions from the previously-coded blocks at different CU sizes. This is a possible direction for our future research.

6) Additionally, we evaluate the complexity-savings from each classifier using 60-frame simulation results under *RDP* setting over 13 standard sequences (described in [37]). The per-classifier results are summarized in Table VIII. From this result, we can see that the complexity is primarily saved from Classifier 1. Relatively, screen content videos (e.g., text and graphics) are more sensitive to *RBFT*. Animation and camera-captured videos behave similarly in our proposed framework. The contribution from Classifier 2 is small, given SCM-4.0 and HM have already embedded fast intra candidate reduction algorithm as described in [15].

7) We compare our proposed ML-FSCC framework with previous SCC fast encoding solutions. Table IX shows that our proposed framework achieves substantially more complexity reductions than the methods of [28], [30] and [32], with only slightly higher BD-rate increase. Compared with [33], the proposed framework achieved a slightly higher complexity reduction, but noticeably higher BD-rate loss. However, the method proposed in [33] relies on stationary region detection in the current frame and reuses the partition side information from the stationary regions within the previous frame. Our proposed framework is self-contained and does not require information from other frames. When the access to the previous frame mode and partition decisions is feasible, the proposed approach can be applied to the non-stationary SC regions, to achieve more complexity savings than both the current approach and [33].

8) We also re-implemented our previous work in [34] onto SCM-4.0 for a fair comparison, as shown in the seventh row

of Table IX. Because the fast partition decision in [34] is purely based on local statistics without necessary IBC search, it incurs higher BD-rate loss than this work.

9) Finally, as mentioned in Section I-C, the previous SCC fast algorithms on each individual mode could be incorporated into our proposed scheme for an additional complexity reduction.

VII. Conclusion and Future Work

In this paper, a novel decision tree based fast CU mode and partition decision framework is proposed for SC compression. The incoming CUs are processed in three steps. Firstly, a block type classifier is used to categorize the incoming CU into either an "NIB" or an "SCB." "NIB" will be encoded using only Intra mode while "SCB" will be encoded using SCC modes at the current CU level. Secondly, the "NIB" is further classified into "Partitioned Block" (P-Block) and "Non-Partition Block" (NP-Block), where P-Block will bypass current level Intra processing and NP-Block will terminate RDOs immediately after the current level Intra processing. Thirdly, NP-Blocks are further classified as either "Directional Blocks" (D-Block) or "Non-Directional Blocks" (ND-Block) and going through a corresponding subset of Intra sub-modes. The trade-off between the encoding efficiency and complexity can be tuned by adjusting the classifier confidence thresholds and rate thresholds. The proposed framework achieved a 40% average complexity reduction with only 1.46% BD-rate loss under "RD-Preserving" setting and yielded a 52% complexity reduction with 3.65% BD-rate loss under "Complexity Reduction Boosting" setting for typical screen content videos under All-Intra configurations. Future studies may generalize the proposed framework to (1) SCC Inter-frame fast coding, (2) online adaptive SC fast encoding and (3) HEVC-SCC fast transcoding applications.

References

[1] G. J. Sullivan *et al.*, "Standardized extensions of high efficiency video coding (HEVC)," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 6, pp. 1001–1016, Dec. 2013.

[2] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1648–1667, Dec. 2012.

[3] R. Joshi *et al.*, *Screen Content Coding Test Model 4 Encoder Description (SCM 4)*, document JCTVC-T1014, Feb. 2015.

[4] C. Pang, J. Sole, L. Guo, M. Karczewicz, and R. Joshi, *Non-RCE3: Intra Motion Compensation With 2-D MVs*, document JCTVC-N0256, Jul. 2013.

[5] J. Chen *et al.*, *Description of Screen Content Coding Technology Proposal by Qualcomm*, document JCTVC-Q0031, Apr. 2014.

[6] L. Guo, M. Karczewicz, and J. Sole, *RCE3: Results of Test 3.1 on Palette Mode for Screen Content Coding*, document JCTVC-N0247, Jul. 2013.

[7] L. Zhang *et al.*, *SCCE5 Test 3.2.1: In-Loop Color-Space Transform*, document JCTVC-R0147, Jun. 2014.

[8] X. Li, J. Sole, and M. Karczewicz, *Adaptive MV Precision for Screen Content Coding*, document JCTVC-P0283, Jan. 2014.

[9] B. Li, J. Xu, G. J. Sullivan, Y. Zhou, and B. Lin, *Adaptive Motion Vector Resolution for Screen Content*, document JCTVC-S0085, Oct. 2014.

[10] C. Pang *et al.*, *CE2 Test1: Intra Block Copy and Inter Signalling Unification*, document JCTVC-T0094, Feb. 2015.

[11] K. Rapaka and J. Xu, *Software for SCM With Hash Based Motion Search*, document JCTVC-Q0248, Mar. 2014.

[12] B. Li and J. Xu, *Hash-Based Motion Search*, document JCTVC-Q0245, Mar. 2014.

[13] L. Zhao, L. Zhang, S. Ma, and D. Zhao, "Fast mode decision algorithm for intra prediction in HEVC," in *Proc. IEEE Vis. Commun. Image Process. (VCIP)*, Tainan, Taiwan, Nov. 2011, pp. 1–4.

[14] W. Jiang, H. Ma, and Y. Chen, "Gradient based fast mode decision algorithm for intra prediction in HEVC," in *Proc. Int. Conf. Consumer Electron., Commun. Netw. (CECNet)*, 2012, pp. 1836–1840.

[15] Y. Piao, J. Min, and J. Chen, *Encoder Improvement of Unified Intra Prediction*, document JCTVC-C207, Jan. 2013.

[16] M. Zhang, J. Qu, and H. Bai, "Entropy-based fast largest coding unit partition algorithm in high-efficiency video coding," *Entropy*, vol. 15, no. 6, pp. 2277–2287, 2013.

[17] X. Shen and Y. Lu, "CU splitting early termination based on weighted SVM," *EURASIP J. Image Video Process.*, vol. 4, pp. 1–11, Dec. 2013.

[18] J. Hou, D. Li, Z. Li, and X. Jiang, "Fast CU size decision based on texture complexity for HEVC intra coding," in *Proc. Int. Conf. Mech. Sci., Elect. Eng. Comput. (MEC)*, 2013, pp. 1096–1099.

[19] H. Zhang and Z. Ma, "Early termination schemes for fast intra mode decision in high efficiency video coding," in *Proc. IEEE ISCAS*, May 2013, pp. 45–48.

[20] H. Zhang and Z. Ma, "Fast intra mode decision for high efficiency video coding (HEVC)," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 4, pp. 660–668, Apr. 2014.

[21] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, no. 4, pp. 438–442, Aug. 1994.

[22] L.-M. Po and W.-C. Ma, "A novel four-step search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 3, pp. 313–317, Jun. 1996.

[23] S. Zhu and K.-K. Ma, "A new diamond search algorithm for fast block-matching motion estimation," *IEEE Trans. Image Process.*, vol. 9, no. 2, pp. 287–290, Feb. 2000.

[24] C.-H. Cheung and L.-M. Po, "A novel cross-diamond search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 12, pp. 1168–1177, Dec. 2002.

[25] C. Zhu, X. Lin, and L.-P. Chau, "Hexagon-based search pattern for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 5, pp. 349–355, May 2002.

[26] A. M. Tourapis, "Enhanced predictive zonal search for single and multiple frame motion estimation," *Proc. SPIE*, vol. 4671, pp. 1069–1079, Jan. 2002.

[27] B. Li and J. Xu, "A fast algorithm for adaptive motion compensation precision in screen content coding," in *Proc. DCC*, Apr. 2015, pp. 243–252.

[28] D.-K. Kwon and M. Budagavi, "Fast intra block copy (IntraBC) search for HEVC screen content coding," in *Proc. IEEE Int. Symp. Circuits Syst.*, Jun. 2014, pp. 9–12.

[29] B. Li, J. Xu, and F. Wu, "A unified framework of hash-based matching for screen content coding," in *Proc. IEEE Vis. Commun. Image Process. (VCIP)*, Dec. 2014, pp. 530–533.

[30] S.-H. Tsang, Y.-L. Chan, and W.-C. Siu, "Fast and efficient intra coding techniques for smooth regions in screen content coding based on boundary prediction samples," in *Proc. ICASSP*, 2015, pp. 1409–1413.

[31] D. Lee, S. Yang, H. J. Shim, and B. Jeon, "Fast transform skip mode decision for HEVC screen content coding," in *Proc. IEEE Int. Symp. Broadband Multimedia Syst. Broadcast. (BMSB)*, Jun. 2015, pp. 1–4.

[32] M. Zhang, Y. Guo, and H. Bai, "Fast intra partition algorithm for HEVC screen content coding," in *Proc. VCIP*, 2014, pp. 390–393.

[33] H. Zhang *et al.*, "Fast intra mode decision and block matching for HEVC screen content compression," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Shanghai, China, Mar. 2016, pp. 1377–1381.

[34] F. Duanmu, Z. Ma, and Y. Wang, "Fast CU partition decision using machine learning for screen content compression," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2015, pp. 4972–4976.

[35] J.-L. Lin, Y.-P. Tsai, Y.-W. Huang, and S. Lei, *Improved Advanced Motion Vector Prediction*, document JCTVC-D125, Jan. 2011.

[36] J. Jung, B. Bross, P. Chen, and W.-J. Han, *Description of Core Experiment 9 (CE9): MV Coding and Skip/Merge Operations*, document JCTVC-609, Jan. 2011.

[37] H. Yu, R. Cohen, K. Rapaka, and J. Xu, *Common Test Conditions for Screen Content Coding*, document JCTVC-T1015, Feb. 2015.

[38] M. Xu, W. Wang, Z. Ma, and H. Yu, *AHG6: Information on the Usage of IBC, Palette, and Intra Prediction in SCC*, document JCTVC-T0194, Geneva, Feb. 2015.

[39] S. G. Blasi *et al.*, "Context adaptive mode sorting for fast HEVC mode decision," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Quebec City, QC, Canada, Sep. 2015, pp. 1478–1482.

[40] E. Peixoto, B. Macchiavello, R. L. de Queiroz, and E. M. Hung, "Fast H.264/AVC to HEVC transcoding based on machine learning," in *Proc. Int. Telecommun. Symp. (ITS)*, 2014, pp. 1–4.
[41] G. F. Escribano *et al.*, "Video encoding and transcoding using machine learning," in *Proc. Int. Workshop Multimedia Data Mining (MDM)*, 2008, pp. 53–62.
[42] Y. Zhang *et al.*, "Machine learning-based coding unit depth decisions for flexible complexity allocation in high efficiency video coding," *IEEE Trans. Image Process.*, vol. 24, no. 7, pp. 2225–2238, Jul. 2015.
[43] L. P. Van, J. De Praeter, G. Van Wallendael, J. De Cock, and R. Van de Walle, "Machine learning for arbitrary downsizing of pre-encoded video in HEVC," in *Proc. IEEE Int. Conf. Consum. Electron. (ICCE)*, Jan. 2015, pp. 406–407.
[44] MATLAB and Statistics Toolbox Release 2013b, MathWorks, Inc., Natick, MA, USA.
[45] J. R. Quinlan. (1998). *C5.0: An Informal Tutorial*. [Online]. Available: http://www.rulequest.com/see5-unix.html
[46] H. Yu *et al.*, *AHG8: New 4:4:4 Screen-Content Sequences for HEVC Extension Development*, document JCTVC-L0301, Jan. 2013.
[47] S. Wang, M. Jiao, T. Lin, and K. Zhou, *AHG8: YUV444 and RGB Screen Content Test Sequences*, document JCTVC-L0317, Jan. 2013.
[48] W. Ding, Y. Shi, and B. Yin, *YUV444 Test Sequences for Screen Content*, document JCTVC-L0317, Apr. 2013.
[49] R. Cohen *et. al.*, *AHG8: 4:4:4 Game Content Sequences for HEVC Range Extensions Development*, document JCTVC-N0294, Aug. 2013.
[50] G. Bjontegaard, *Calculation of Average PSNR Differences Between RD Curves (VCEG-M33)*, document ITU-T SG16 Q.6, VCEG Meeting, Austin, TX, USA, Apr. 2001.

Fanyi Duanmu received the B.S. and M.S. degrees from the Beijing Institute of Technology, Beijing, China, and Polytechnic Institute of New York University, Brooklyn, NY, USA, in 2009 and 2011, respectively. He is currently working toward the Ph.D. degree in electrical and computer engineering at New York University, Tandon School of Engineering, Brooklyn, NY, USA.

His current research focuses on video compression and streaming, the next-generation video coding, machine learning driven video processing and computer vision.

Zhan Ma received the B.S. and M.S. from Huazhong University of Science and Technology (HUST), Wuhan, China, in 2004 and 2006 respectively, and the Ph.D. degree from Polytechnic School of Engineering, New York University (formerly Polytechnic University, Brooklyn, NY), Brooklyn, NY, USA, in 2011.

He is now on the faculty of Electronic Science and Engineering School, Nanjing University, Jiangsu, China. From 2011 to 2014, he has been with Samsung Research America, Dallas TX, USA, and FutureWei Technologies, Inc., Santa Clara, CA, USA, respectively. His current research focuses on the next-generation video coding, energy-efficient communication, and multi-spectral signal compression.

Yao Wang (F'04) received the B.S. and M.S. in electronic engineering from Tsinghua University, Beijing, China, in 1983 and 1985, respectively, and the Ph.D. degree in electrical and computer engineering from University of California at Santa Barbara, Santa Barbara, CA, USA, in 1990.

Since 1990, she has been on the faculty of Electrical and Computer Engineering, Tandon School of Engineering of New York University (formerly Polytechnic University, Brooklyn, NY, USA). Her current research areas include video communications, multimedia signal processing, and medical imaging. She is the leading author of a textbook titled "Video Processing and Communications," and has published over 250 papers in journals and conference proceedings.

Dr. Wang has served as an Associate Editor for IEEE Transactions on Multimedia and IEEE Transactions on Circuits and Systems for Video Technology. She received New York City Mayor's Award for Excellence in Science and Technology in the Young Investigator Category in year 2000. She was elected Fellow of the IEEE in 2004 for contributions to video processing and communications. She is also a co-winner of the IEEE Communications Society Leonard G. Abraham Prize Paper Award in the Field of Communications Systems in 2004, and a co-winner of the IEEE Communications Society Multimedia Communication Technical Committee Best Paper Award in 2011. She was a keynote speaker at the 2010 International Packet Video Workshop.