Complexity reduction of VVC video encoding using machine learning techniques

Alexandre Tissier

DOCTORAT BRETAGNE LOIRE / MATHSTIC

INSA RENNES

THÈSE DE DOCTORAT DE

INSA DE RENNES

ÉCOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies de l'Information et de la Communication
Spécialité : *Signal, Image, Vision*

Par

Alexandre TISSIER

Complexity reduction of VVC video encoding using machine learning techniques

Thèse présentée et soutenue à Rennes, le 25 mai 2022
Unité de recherche : Unité de recherche : IETR
Thèse N° : 22ISAR 10 / D22 - 10

Rapporteurs avant soutenance :

MOKRAOUI Anissa	Professeur, Université Sorbonne Paris Nord
CAGNAZZO Marco	Associate Professor, Padua University

Composition du Jury :

Président :	COUDOUX François-Xavier	Professeur, Univ. Polytechnique Haut de France
Examinateurs :	MOKRAOUI Anissa	Professeur, Université Sorbonne Paris Nord
	CAGNAZZO Marco	Associate Professor, Padua University
	HAMIDOUCHE Wassim	Maitre de conférences, INSA Rennes
	VANNE Jarno	Associate Professor, Tampere University
Dir. de thèse :	MENARD Daniel	Professeur, INSA Rennes

Table des matières

I Background **3**

1 Introduction **5**
1.1 Challenges and objectives . . . 6
1.2 Contributions . . . 8
1.3 Thesis Outline . . . 10

2 Video coding background **11**
2.1 Video characteristics . . . 11
2.1.1 Color space . . . 11
2.1.2 Spatial resolution . . . 12
2.1.3 Temporal resolution . . . 12
2.1.4 Bit depth . . . 13
2.2 Quality and complexity metrics . . . 13
2.2.1 Objective quality metrics . . . 14
2.2.1.1 Full reference methods . . . 14
2.2.1.2 Reduced and no reference methods . . . 15
2.2.2 Subjective quality metrics . . . 16
2.2.3 Complexity reduction metric . . . 16
2.3 Versatile Video Coding test model and Common Test Conditions . . . 16
2.4 Video coding standards . . . 18
2.5 Versatile Video Coding . . . 19
2.6 Encoding tools . . . 21
2.6.1 Tree partitioning . . . 21
2.6.1.1 Coding Tree Unit . . . 21
2.6.1.2 Slice and tile . . . 21
2.6.1.3 Coding Unit . . . 22
2.6.2 Intra mode . . . 24
2.6.3 Inter mode . . . 25
2.6.4 Transform . . . 27
2.6.5 Quantization . . . 28
2.6.6 In-loop filters . . . 29
2.6.7 Entropy encoding . . . 30
2.6.8 Encoder configurations . . . 30
2.7 Rate-Distortion Optimization . . . 31

2.8 Progression of Versatile Video Coding test model coding efficiency and complexity . . . 32

3 Related works 35
3.1 Introduction . . . 35
3.2 State of the art complexity reduction techniques for High Efficiency Video Coding . . . 36
3.2.1 Complexity reduction of the tree partitioning . . . 37
3.2.1.1 Handcrafted methods . . . 37
3.2.1.2 Decision Tree methods . . . 38
3.2.1.3 Convolutional Neural Network methods . . . 39
3.2.2 Complexity reduction of prediction modes . . . 40
3.3 State of the art complexity reduction techniques for Future Video Coding . 41
3.3.1 Complexity reduction of the tree partitioning . . . 41
3.3.1.1 Handcrafted methods . . . 41
3.3.1.2 Decision Tree methods . . . 42
3.3.1.3 Convolutional Neural Network methods . . . 44
3.4 State of the art complexity reduction techniques for Versatile Video Coding 45
3.4.1 Complexity reduction for tree partitioning . . . 46
3.4.1.1 Handcrafted methods . . . 47
3.4.1.2 Decision Tree methods . . . 48
3.4.1.3 Convolutional Neural Network methods . . . 49
3.4.2 Complexity reduction of intra mode prediction and Multiple Transform Selection . . . 51
3.5 Complexity reduction of coding tools . . . 52
3.5.1 Transform . . . 52
3.5.2 Quantization . . . 52
3.5.3 Parallelization . . . 52
3.6 Real-time Versatile Video Coding encoders and decoders . . . 53
3.6.1 Study and directions for real-time Versatile Video Coding encoding . 53
3.6.2 Practical Versatile Video Coding encoder . . . 53
3.6.3 Real-time Versatile Video Coding decoder . . . 54
3.7 Machine Learning in video coding . . . 54
3.7.1 End-to-end video coding frameworks . . . 54
3.7.2 Video coding tools exploiting Machine Learning . . . 55
3.7.2.1 Intra prediction mode . . . 55
3.7.2.2 Inter prediction mode . . . 56
3.7.2.3 In-loop filter . . . 56
3.7.3 Super-resolution pre and post-processing . . . 57

II Contributions 59

4 Complexity reduction opportunities in the practical Versatile Video Coding encoder 61
4.1 Complexity analysis of the Versatile Video Coding test model encoder . . . 62
4.1.1 Impact of resolution on complexity . . . 62
4.1.2 Impact of Quantization Parameter on complexity . . . 63
4.1.3 Impact of tree partitioning on the encoder complexity . . . 64
4.1.4 Impact of split depth on the encoder complexity . . . 66

4.2 Complexity reduction opportunity in the Versatile Video Coding test model encoder . . . 68
4.2.1 Determination of the complexity reduction opportunities . . . 68
4.2.2 Analysis of complexity reduction opportunities for Versatile Video Coding test model 3.0 under All Intra configuration . . . 69
4.2.3 Complexity reduction opportunities for the Versatile Video Coding test model 10.2 under All Intra and Random Access configurations . 71
4.3 Conclusion . . . 73

5 Deep learning based complexity reduction of Versatile Video Coding intra encoder 75
5.1 Introduction . . . 75
5.2 Intermediate features extraction with Convolutional Neural Network . . . 75
5.2.1 Overall method . . . 76
5.2.2 Convolutional Neural Network structure . . . 77
5.3 Convolutional Neural Network training . . . 78
5.3.1 Dataset for training . . . 78
5.3.2 Dataset optimisation . . . 80
5.3.3 Training process description . . . 80
5.4 Experimental results . . . 81
5.4.1 Convolutional Neural Network performance . . . 81
5.5 Convolutional Neural Network inference optimisation . . . 83
5.5.1 Optimisation on Central Processing Unit . . . 83
5.5.2 Optimisation on Graphics Processing Unit . . . 84
5.6 Conclusion . . . 85

6 Threshold based partitioning scheme for Versatile Video Coding intra encoders 87
6.1 Introduction . . . 87
6.2 Partitioning prediction based on a Convolutional Neural Network . . . 87
6.2.1 Threshold-based decision . . . 88
6.2.2 Threshold analysis . . . 89
6.2.3 Threshold selection . . . 93
6.3 Experimental results . . . 95
6.3.1 Experimental setup . . . 95
6.3.2 Complexity reduction performance under the Versatile Video Coding test model . . . 95
6.4 Conclusion . . . 98

7 Machine learning based Quad-Tree-Multi-Type Tree partitioning scheme for Versatile Video Coding intra encoders 99
7.1 Introduction . . . 99
7.2 Partitioning prediction based on Machine Learning . . . 100
7.2.1 Overall presentation . . . 100
7.2.2 Multi-class classifiers based on Decision Tree models . . . 101
7.2.3 Training process for Decision Tree LightGBM model . . . 101
7.3 Experimental results . . . 102
7.3.1 Experimental setup . . . 102
7.3.2 Decision Tree LightGBM performance . . . 102

7.3.3 Complexity reduction performance under the Versatile Video Coding test model . . . 104
7.3.4 Complexity overhead . . . 107
7.4 Conclusion . . . 108

8 Machine learning based Quad-Tree-Multi-Type Tree partitioning for Versatile Video Coding inter coding 109
8.1 Introduction . . . 109
8.2 Proposed method . . . 110
8.2.1 Overall presentation . . . 110
8.2.2 Convolutional Neural Network . . . 112
8.2.3 Multi-Class Classifier . . . 112
8.3 Dataset . . . 113
8.3.1 Dataset description . . . 113
8.3.2 Dataset pre-processing . . . 114
8.4 Training process and experimental setup . . . 115
8.4.1 Convolutional Neural Network training parameters . . . 115
8.4.2 Multi-Class Classifier training parameters . . . 116
8.4.3 Experimental setup . . . 117
8.5 Experimental results . . . 117
8.5.1 Convolutional Neural Network performance . . . 117
8.5.2 Multi-Class Classifier performance . . . 119
8.5.3 Complexity reduction and coding performance . . . 119
8.6 Conclusion . . . 124

III Conclusion 125

9 Conclusion 127
9.1 Complexity reduction opportunities in the practical Versatile Video Coding encoder . . . 127
9.1.1 Key results . . . 127
9.2 Threshold based partitioning scheme for Versatile Video Coding intra encoders 128
9.2.1 Key results . . . 128
9.2.2 Future works . . . 128
9.3 Machine learning based Quad-Tree-Multi-Type Tree partitioning scheme for Versatile Video Coding intra encoders . . . 129
9.3.1 Key results . . . 129
9.3.2 Future works . . . 129
9.4 Machine learning based Quad-Tree-Multi-Type Tree partitioning for Versatile Video Coding inter coding . . . 129
9.4.1 Key results . . . 129
9.4.2 Future works . . . 130
9.5 Other perspectives . . . 130

A French summary 133
A.1 Challenges et objectifs . . . 134
A.2 Contributions . . . 135
A.3 Plan du manuscript . . . 137

List of Figures **141**

List of Tables **144**

Personal Publications **145**

Bibliography **153**

Acknowledgements

Merci à Daniel, Wassim et Jarno de m'avoir encadré. Merci de la confiance que vous m'avez accordé ainsi que l'autonomie dont j'ai disposé. Vos réflexions ainsi que l'aide que vous m'avez donné lors de la rédaction des articles ont été très important pour le bon déroulement de ma thèse.

Merci aux membres de mon jury d'avoir pris le temps de relire mon manuscrit ainsi que d'avoir assisté à ma soutenance. Vos questions lors de la soutenance ont été très intéressantes et enrichissantes.

Merci à Vincent, Cédric et Souhaiel, les stagiaires qui m'ont grandement aidé durant ma thèse grâce à leur bonne volonté et leur bonne humeur.

Merci aux copains du labo que j'ai pu me faire tout au long de ma thèse. J'étais bien heureux de venir à l'INSA tous les jours pour voir vos têtes. Et j'étais encore plus heureux de partir en vacances, week-ends, soirées, apéros avec vous.

Merci à tous mes copains que j'ai vu en dehors de la thèse qui m'ont bien détendu grâce à nos sacrées mixtures. Les vacances et week-ends avec vous sont toujours des grands moments de joie.

Merci à ma famille qui m'a toujours soutenu et poussé dans mes projets. Sans vous je n'aurais pas réussi à faire tout ce que j'ai fait dans ma vie.

Je vous aimes tous, des bisous.

Kiki

Première partie

Background

CHAPITRE 1

Introduction

The exploitation opportunities of media applications have been revolutionized in the last decade with the emergence of new services like video on demand, video streaming, or video sharing platforms. Furthermore, the innovations brought to the data communication domain has also transformed the amount of transmitted data. Indeed, the 4G, 5G technologies, and optical fiber have allow a significant increased in bandwidth and are thereby better able to cope with the video data explosion.

Video represents the vast majority of global data traffic according to Cisco's study [1] in 2019. The evolution of the global IP traffic across the different application categories is depicted in Figure 1.1 from 2017 to 2022. In 2017, the global IP traffic was more than 100 exabytes per month and Cisco predicted that the IP traffic will reach almost 400 exabytes per month in 2022. The total amount of the global IP traffic is almost quadrupled in 5 years.

The category with the most impact on the global IP traffic is the internet video with more than 60 exabytes per month in 2017 and is predicted to be 275 exabytes per month in 2022. The internet video category took more than half of the whole IP traffic in 2017 and will continue to grow, with 71% in 2022. Furthermore, the IP VOD also takes an important part of the whole IP traffic with 20% in 2017 to 11% in 2022. On average, 82% of the whole IP traffic (315 exabytes per month) is dedicated to video transmission.

The emerging video formats such as 8K, High-frame rate (HFR), 360° video, multiview, light field, and point clouds increase the quality of experience through the resolution, frame rate, or immersion but at the expense of significant bandwidth increase. For example, an 8K Ultra High Definition (UHD) image contains 33 177 600 pixels which is 96× that of Standard Definition (SD) (720×480). More specifically, the Figure 1.2 details the internet video traffic from 2017 to 2022 separated by three resolutions : SD, High Definition (HD) and UHD. In 2017, the internet video traffic was shared between SD and HD resolutions. In 2022, UHD resolution was predicted to 22% of the global video traffic and that the SD will drop to 21%. HD resolution video keeps the same proportion between the categories but the transmitted video data is quadrupled between 2017 and 2022.

The emerging video formats alongside with the explosion of IP video traffic [1] require new video coding techniques that outperform existing ones. Organizations such as ISO/IEC, ITU-T Joint Video Expert Team (JVET) and Alliance for Open Media (AOM) specify new video coding standards. Most recently, AOM developed the AV1 codec released in 2018 as a successor to VP9 and JVET standardized Versatile Video Coding (VVC)

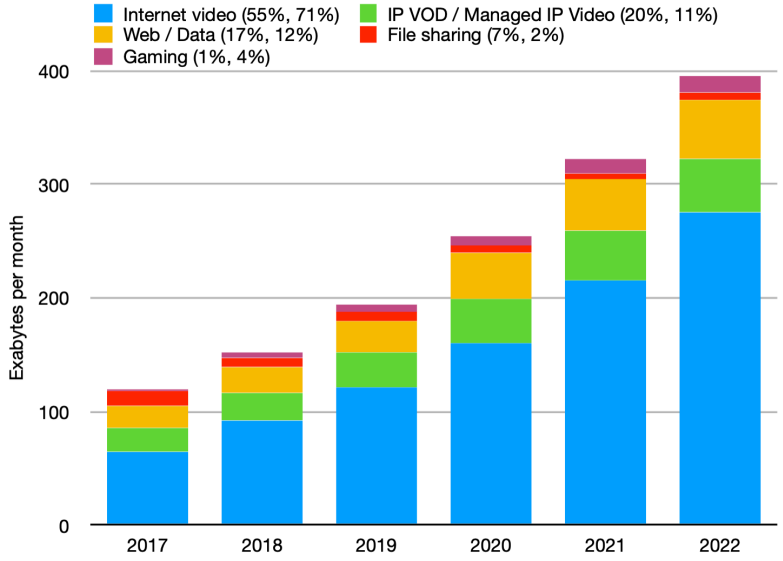


Figure 1.1 – *Global IP traffic in exabytes per month from 2017 to 2022 by application categories.*

ITU-T H.266 | MPEG-I - Part 3 (ISO/IEC 23090-3) [2] in July 2020 as a successor to High Efficiency Video Coding (HEVC).

1.1 Challenges and objectives

The objective of the video coding is to minimize a trade-off between rate and distortion, named Rate-Distortion (RD)-cost, i.e., to obtain the minimum bit rate for a given quality or the maximum quality for a given bit rate. All international video coding standards share the same hybrid video coding structure. Therefore, during standardization different coding tools were integrated to improve the intra and inter predictions, transform, or to enhance the block partitioning process. The selection of specific coding tools leads to different coding efficiencies and computational costs. Comparative studies were conducted between VVC and AV1 [3], [4] based on subjective and objective quality assessment metrics including Peak Signal-to-Noise Ratio (PSNR) and Video Multimethod Assessment Fusion (VMAF). VMAF is a Machine Learning (ML) based quality metric relying on detail loss metric, visual information fidelity measure, and averaged temporal frame difference. Both works conclude that VVC outperforms AV1 in terms of both objective and subjective quality scores. However, computational cost of AV1 is slightly lower than that of VVC and varies between the coding configurations.

The VVC reference software, called VVC Test Model (VTM), implements all normative VVC coding tools allowing rate-distortion-complexity evaluation and conformance testing. VTM brings 25.32% and 36.95% bitrate reductions at the expense of significant increases in encoder computational complexity of 2630%(x27) and 859%(x9) over the HEVC test Model (HM) 16.22 [5] in All Intra (AI) and Random Access (RA) configurations, respectively. These coding gains are even higher for HD and UHD video sequences [6].

The encoding process aims at solving the Rate-Distortion Optimization (RDO) problem which corresponds to a combinatorial optimization process. The encoder must select the coding parameters, leading the minimal RD-cost. Currently, the VTM encodes a block

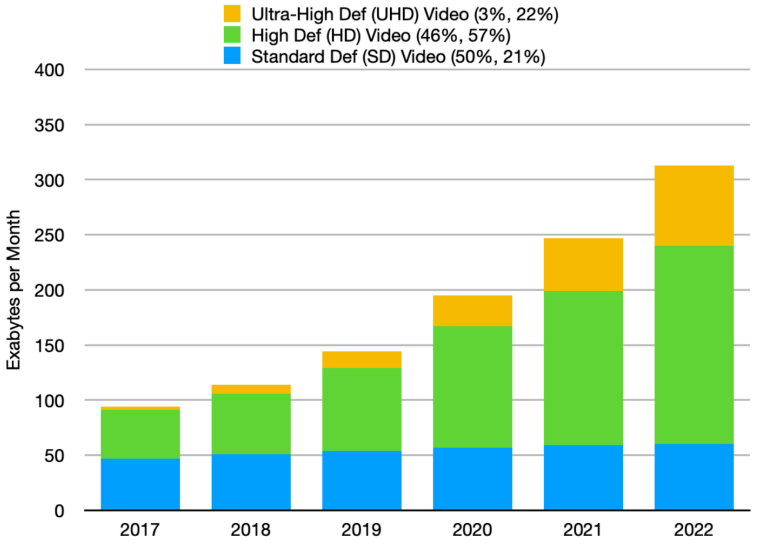


Figure 1.2 – *Internet video traffic in exabytes per month from 2017 to 2022 by resolution categories.*

of pixels with all possible configurations of coding tools and retains the one leading the best RD-cost. This exhaustive test results in prohibitive encoding times. The search space of this RDO problem has been significantly extended compared with HEVC, due to the integration of new coding tools and the extension of the search space of existing tools. As mentioned in [7], the tree partitioning scheme is the most efficient of the new VTM coding tools but it also leads to the highest complexity increase.

Compared with HEVC, which is based on a Quad-Tree (QT) block partitioning, VVC integrates a nested Multi-Type Tree (MTT) partitioning scheme allowing in addition to QT, horizontal and vertical Binary-Tree (BT) and Ternary-Tree (TT) splits [8]. BT and TT allow a block of pixels to be rectangular instead of only square which enables the block to better fit the characteristics of the frame. At each level of the hierarchical tree partitioning process, up to five splits are tested by the encoder, compared with one split, i.e., QT split, in HEVC. Authors in [9] have shown that disabling BT and TT splits decreases the encoding time by 91.7%. Therefore, the tree partitioning process offers an high opportunity for complexity reduction.

The objective of our research work in this thesis was to reduce the complexity of the VVC encoder while maintaining high coding efficiency. To achieve real-time VVC encoding, several tools need to be modified and especially their exhaustive searches. One of the key aspects is the complexity of the QT-MTT partitioning process which must be significantly reduced. Indeed, this process has the most complexity reduction opportunity among the VVC tools. Our work will pave the way for VVC complexity reduction through the tree partitioning process optimization. Furthermore, our work aims to give directions for real-time encoder to prune the tree partitioning by maximizing both the complexity reduction and the encoding quality loss.

1.2 Contributions

This thesis details the four contributions that study and reduce the complexity of the VTM encoder. The first contribution investigates the VTM complexity reduction opportunities depending on the encoding tools. The three other contributions seek to reduce the complexity of the VTM encoder through ML techniques. These three contributions focus on maximizing the complexity reduction while minimizing the coding quality loss of the encoder. The contributions are proposed during the VVC standardisation. They are one of the first complexity reduction techniques and will be interesting as a basis for comparison. Furthermore, as VVC brings new tools with high complexity, our techniques are relevant for the professional encoders that need to highly reduce their complexity. Indeed, those contributions are intended to inspire companies and developers of practical encoders. The four contributions proposed in this thesis are briefly presented below.

I Complexity reduction opportunities in the practical Versatile Video Coding encoder

Complexity reduction has been a hot topic in video coding since HEVC. VVC brings impressive coding quality compared with HEVC but at the cost of high encoding complexity increase. Indeed, through the VVC standardization, JVET has investigated many new coding tools such as the MTT partitioning, 32 new angular intra prediction modes, and 2 new transforms. In this context, a hierarchical characterization and evaluation of the VVC encoding complexity with the VTM encoder is assessed. First, an overview is presented on the complexity impact of encoding parameters. The studied parameters include spatial resolution, Quantization Parameter (QP), and the split impact. Then, the complexity reduction opportunities offered by three encoding levels are defined and evaluated with : the tree partitioning, the intra prediction mode, and the Multiple Transform Selection (MTS) process under the VTM3.0 in AI configuration. With the VTM10.2, the tree partitioning opportunity is assessed and evaluated under both AI and RA configurations. This study demonstrates that the relative complexity of VTM encoding is proportional to video resolution and QP. It also shows that the splits have an important impact on the VTM encoding complexity and Bjøntegaard Delta Bit Rate (BD-BR) loss.

Furthermore, the main contribution was to study the theoretical upper limits of various complexity reduction techniques. In the beginning of this thesis this will be used to identify interesting tools to focus on. The tree partitioning is shown to have potential for complexity reduction of up to 97% whereas the respective percentages for intra mode prediction and MTS are 65% and 55% with VTM3.0 under AI configuration. The tree partitioning complexity reduction opportunity for the VTM10.2 has the same results as that for VTM3.0 under AI configuration. Under RA configuration for the tree partitioning process, the complexity reduction opportunity is up to 91.2% which is close to the AI opportunity. Furthermore, based on these first results, a more specific study is proposed on the splits and their depths which will be useful for the next contributions.

II Threshold based partitioning scheme for Versatile Video Coding intra encoders

This contribution focuses on the complexity reduction of the VTM encoder under AI configuration. An efficient Convolutional Neural Network (CNN)-based technique is proposed to reduce the complexity of the VTM6.1 intra encoder. A CNN is fed with a 64×64 pixels luminance Coding Unit (CU) and it predicts a vector that gives probabilities of having

edges at the 4×4 boundaries of the block. From the probability of boundaries, a split probability is deduced and compared with a threshold to skip unlikely splits. Two solutions are proposed to determine the threshold including uniform and non-uniform. The non-uniform threshold solution makes the thresholds evolve through the split categories and the CU sizes relying on the wanted trade-offs between complexity and quality loss. In VTM6.1 intra coding, the proposed solution enables 42.2% complexity reduction for a slight BD-BR increase of 0.75%. With high-resolution sequences, the speed-up is even higher, up to 54.5% complexity reduction at a cost of 0.85% BD-BR loss. The non-uniform threshold solution reaches 54.5% complexity reduction for 1.33% BD-BR loss.

III Machine learning based Quad-Tree-Multi-Type Tree partitioning scheme for Versatile Video Coding intra encoders

In this contribution, the predicted probabilities are treated as spatial features. Indeed, an efficient technique is proposed to reduce the complexity of the QT-MTT partitioning. Our approach combines a moderate complexity CNN that extracts spatial features and multi-class classifiers that derive the best splits to be tested in the tree partitioning process. This CNN predicts the probability of each boundary of all the 4×4 pixel blocks within the input 64×64 block. At each level of the hierarchical tree partitioning process, a multi-class classifier is used to predict from the set of boundary probabilities, the N-most likely splits to explore by the encoder. One classifier is trained for each size of the 16 different sub-blocks. At each depth, the number of tested partitions N can be adjusted from one to the total number of possible splits. This allows exploring wide range of trade-offs between the complexity reduction and the quality loss. The proposed solution with top-$N = 3$ configuration reaches 46.6% complexity reduction for a negligible BD-BR increase of 0.86%. The top-$N = 2$ configuration enables on average a complexity reduction of 69.8% for 2.57% BD-BR loss.

IV Machine learning based Quad-Tree-Multi-Type Tree partitioning for Versatile Video Coding inter coding

This contribution focuses on the complexity reduction under the RA configuration which modifies the tree partitioning selection. An efficient complexity reduction method that relies on ML algorithm is proposed. First, a CNN predicts the partition based on its $128 \times 128 \times 3$ luminance pixel values. The input is the current Coding Tree Unit (CTU) processed plus the two reference CTUs obtained by the inter mode prediction process. These two reference CTUs provide the temporal information used to deduce the partition search. The output vector represents the partition of the current CTU. For more accurate decision, Decision Tree (DT)s are exploited at each depth levels. The tree partitioning information predicted by the CNN is given as input to the DTs depending on the block depth and size. Several DTs are available to predict the probabilities of each split such as a multi-classification task. At each CU size, a DT is accessible as the input is different from one another. DT models are able to predict the split probabilities from 128×128 to 4×8, or 8×4 block sizes. Finally, relying on the targetted complexity reduction or BD-BR, a number of split skipped is defined. The highest split probabilities are tested inside the VTM to limit the BD-BR loss and maximize the complexity reduction. Our $C1$ solution brings 31.8% complexity reduction for 1.11% BD-BR loss on the Common Test Conditions (CTC).

1.3 Thesis Outline

This section summarizes the content of each chapter included in this thesis. Chapter 2 presents the overview of video coding which is necessary to understand the different contributions proposed in this thesis. The video coding standards are introduced and especially the VVC standard with its new tools and the RDO that optimizes its selections during the process. Furthermore, the CTC and metrics that allow to compare the results fairly are specified.

Chapter 3 details the state of the art techniques that reduce the complexity of different standards. First, the complexity reduction techniques for HEVC standard are presented, followed by the techniques performed for Future Video Coding (FVC). FVC is an experimental standard of JVET, which contains coding tools that were studied for achieving coding performance beyond that of HEVC. The VVC complexity reduction techniques are depicted focusing on the tree partitioning, intra mode prediction and MTS process. A brief presentation is proposed on techniques that reduce the complexity of other tools. Real-time encoder and decoder are also described. Finally, an introduction on machine learning-based techniques is proposed focusing on video coding.

Chapter 4 describes the study on the complexity reduction opportunities. An analysis is first proposed to determine the complexity impact through several parameters. Then, the complexity reduction opportunities are evaluated for the tree partitioning, intra mode prediction, and MTS processes under AI and RA configurations.

A CNN is described in Chapter 5 which is the basis of the next two chapters. The training method and parameters are detailed with the CNN performance. Furthermore, the CNN inference complexity is studied and optimize to determine its impact on the encoding time. The fastest configuration with 16-bit floating-point data reaches 16.2 Frames per Second (FPS) at full HD resolution on the Nvidia GTX 1650 Max Q Graphics Processing Unit (GPU).

A first complexity reduction solution is described in Chapter 6 along with its results. This method focuses on the previous CNN that predicts the tree partitioning based on the luminance pixels of the considered block. The block partitioning is defined as a vector that represents the whole block partition through probabilities. A comparison between those probabilities and a threshold enables skipping unlikely splits. Two approaches for threshold determination are proposed with a uniform threshold and a non-uniform threshold solutions. The first complexity reduction solution reaches 54.5% complexity reduction for 1.33% BD-BR loss.

Chapter 7 details our method that exploits the features predicted by the CNN with a DT to decide the split to test. The CNN output is considered as features to optimize the probability distribution. Those features are given to a DT model that predicts the split probabilities. The used dataset is presented with its optimisation to enhance the ML results. Different configurations are available based on the N split performed inside the encoder. This method halves the complexity reduction for less than 1% BD-BR loss.

Chapter 8 seeks to reduce the complexity for inter coding under the RA configuration. First, a CNN is trained to predict a map probability. This output is based on the luminance pixels of the CTU computed plus its two reference CTUs obtained by the most related Motion Vector (MV)s. Then, the map probability is given to the DT that predicts the split probabilities. As in previous solution, the N highest split probabilities are computed to maximise the complexity reduction while minimising the BD-BR loss. This solution, under RA configuration, reaches 31.8% complexity reduction for 1.11% BD-BR loss.

Finally, Chapter 9 concludes this thesis and gives several future research directions.

CHAPITRE 2

Video coding background

This chapter describes the video coding background through different aspects. First, a global presentation of the video characteristics is proposed in the Section 2.1. Section 2.2 describes the different video coding metrics. The Versatile Video Coding (VVC) Common Test Conditions (CTC) are detailed in Section 2.3. Section 2.4 presents the video compression standards developed across the years. The global scheme of the VVC encoder and its corresponding tools are introduced in Section 2.5 and in Section 2.6. Section 2.7 explains the principle of the Rate-Distortion Optimization (RDO) which is performed in order to select the optimal configuration of the different coding tools. Finally, Section 2.8 presents the encoding and decoding complexity plus the Bjøntegaard Delta Bit Rate (BD-BR) gain through the VVC Test Model (VTM) version during the VVC standardization against the previous standard High Efficiency Video Coding (HEVC).

2.1 Video characteristics

A video is composed of a succession of frames which are themselves defined by a set of pixels. A video is characterized by different parameters which are described in the rest of this section.

2.1.1 Color space

Color video are mainly represented in two formats RGB or YUV. For the YUV format, each pixel is composed of three values, one of luminance (Y) that represents the luminous intensity and two of chrominance (U and V) that represent the color information. The RGB is an additive color model in which red, green, and blue are added together with different intensity levels to obtain a broad range of color. A conversion is available between the two formats through these equations :

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.14713 & -0.28886 & 0.436 \\ 0.615 & -0.51498 & -0.10001 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}, \tag{2.1}$$

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.13983 \\ 1 & -0.39465 & -0.58060 \\ 1 & 2.03211 & 0 \end{bmatrix} \begin{bmatrix} Y \\ U \\ V \end{bmatrix}. \tag{2.2}$$

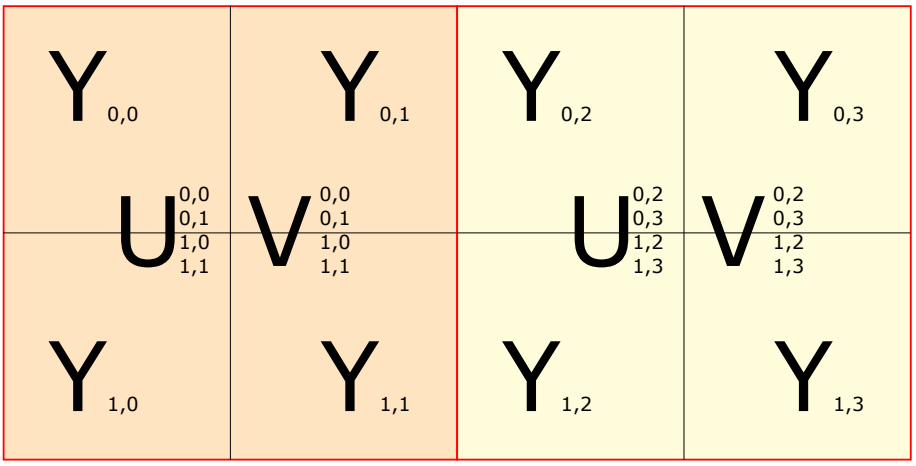


Figure 2.1 – *Representation of YUV* 4 : 2 : 0.

Eq. 2.1 transforms the RGB values of a pixel to YUV values and Eq. 2.2 transforms the YUV values to RGB values.

In video coding, the input and output video format is YUV. YUV is compatible for black-and-white television and color television due to the luminance and chrominance separation which allows the transmission of the luminance (Y) only for grey scale television. Furthermore, the chrominance takes advantage of the human visual system that has lower acuity for color differences than luminance. It results in different chrominance subsamplings defined by a three part ratio $J : a : b$ that describes the number of luminance and chrominance samples in a conceptual region of J pixels wide and 2 pixels high, with J the horizontal sampling reference which is usually 4, a the number of chrominance samples (U and V) in the first row of J pixels and b the number of changes of chrominance samples between first and second row of J pixels. The chrominance subsampling used in this thesis is $4 : 2 : 0$, i.e., one value of U and V for four values of Y, such as presented in Figure 2.1. $Y_{x,y}$, $U_{x,y}$, and $V_{x,y}$ are the luminance and chrominance values of the pixel at the coordinate (x, y). This video format allows a first compression of the video by taking into consideration only 1 out of 4 chrominance values.

2.1.2 Spatial resolution

A video is also characterized by its spatial resolution $W \times H$ pixels. The term W defines the number of pixel columns (width) and H is the number of pixel rows (height). A frame can be represented as a two dimensional matrix of size $W \times H$ and each pixel is accessed through coordinates (x, y), with x the column number and y the row number of the pixel location inside the frame. Figure 2.2 shows several resolution from SD to 8K Ultra High Definition (UHD). Increasing the resolution results in more details for the same video content.

2.1.3 Temporal resolution

A video is a sequence of frames displayed at a certain rate. This frame-rate is characterized by the metric Frames per Second (FPS) which is the frequency at which consecutive frames are shown on a display. It goes from 24 FPS which is the minimum speed needed while maintaining realistic motion to more than 60 FPS for athletic or high-motion video for instance. Figure 2.3 shows the same video with a duration of 100ms at 60 FPS and 24

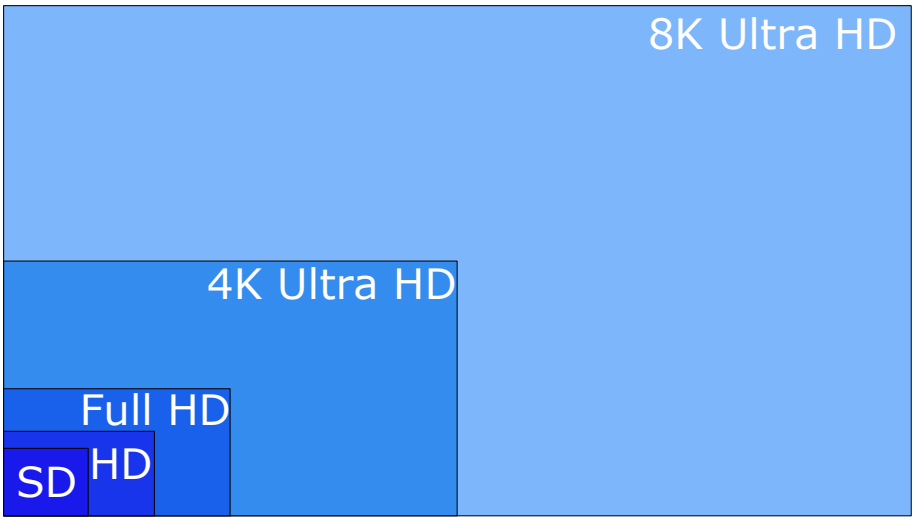


Figure 2.2 – *Spatial resolutions from SD (720x480) to 8K Ultra HD (7680x4320).*

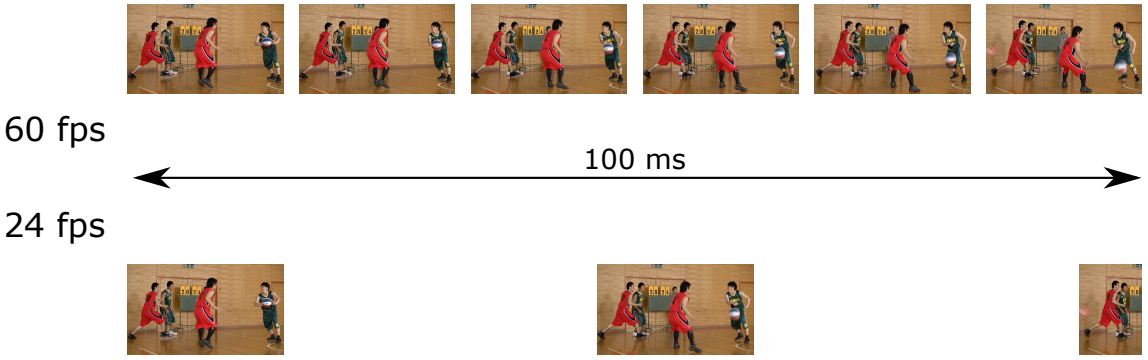


Figure 2.3 – *Example of the same video in 60 FPS and 24 FPS for 100ms.*

FPS. Having higher frame-rate increases the motion consistency in the video and limits the motion related artefacts such as motion blur.

2.1.4 Bit depth

In order to increase the precision of the pixels, the pixels are stored and processed at different bit depth. Under the coding process, the luminance and the two values of chrominance are stored for input and output video in 8 or 10 bits. With 8 bits, Y, U, and V values $\in [0, 255]$, which allows 16 277 216 pixel values (256^3). A bit depth of 10 bits increases the pixel dynamic range with 1 073 741 824 values (1024^3) as Y, U, and V values $\in [0, 1023]$.

2.2 Quality and complexity metrics

Figure 2.4 presents the process of a video from the video acquisition until the video delivery. After the acquisition, the video is compressed with an encoder which creates a bitstream. Then the decoder interprets the bitstream to generate a decoded video. The bitstream is evaluated with its rate which defines the number of bits transmitted per second. The most commonly used quality metric between the encoded video and the decoded video is the Peak Signal-to-Noise Ratio (Psnr) which is based on the Mean-Squared Error (MSE) that represents the distortion between those two videos. A BD-BR [10] is deduced through

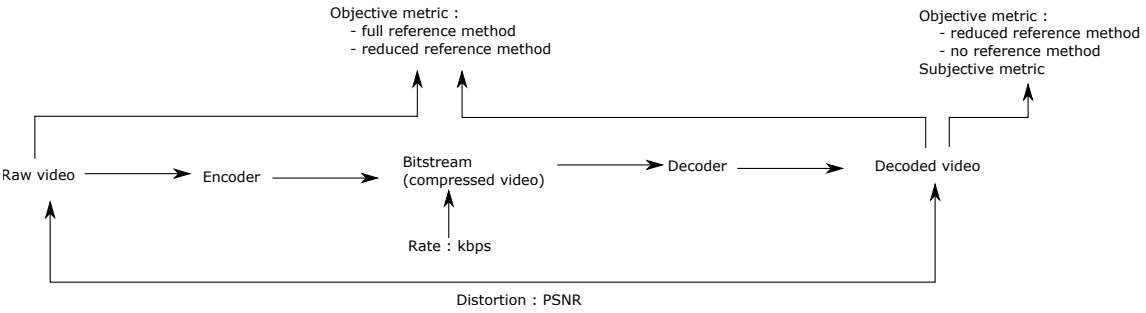


Figure 2.4 – *Encoding and decoding process related to metrics.*

the bitrate and the Psnr to evaluate the coding efficiency between two rate-distortion curves. In order to assess the coding performance, several quality metrics are proposed. Two distinct branches are available with the objective and subjective metrics which are performed by numerical criteria or explicit criteria, respectively. Some objective methods are based on the comparison between the raw video and the decoded video, others only refer to the decoded video due to the lack of raw video information (blind). The metrics are processed to evaluate the quality of different video codecs, encoders, encoding settings, or transmission variants.

2.2.1 Objective quality metrics

Three categories can be considered for objective metrics : the full reference method, the reduced reference method, and the no reference method. The full reference method computes the quality difference by comparing the raw video signal which is uncompressed against the evaluated decoded video signal. With this type of method, typically, every pixel from the raw sequence is compared against the corresponding pixel at the same location of the decoded sequence, without any knowledge from the encoding or transmission process that happens between those two sequences. The reduced reference method extracts features of both videos and compare them to give a quality score while the original video is not always fully available. This method is performed when the raw video is not fully accessible. Then, the no reference method assesses the quality of the decoded video without any information on the raw video or its features.

2.2.1.1 Full reference methods

The most widely used objective metric in video coding is the Psnr, it defines the distortion between two images. However, this image quality metric performs poorly when focusing on the human perception. Indeed, it is only based on the pixel differences which is simple to implement and fast to compute compared with other quality metrics. The Psnr is defined via the MSE that corresponds to the power of the corrupting noise of the image evaluated compared with the reference image :

$$MSE = \frac{1}{W\,H} \sum_{i=0}^{W-1} \sum_{j=0}^{H-1} [I_r(i,j) - I_e(i,j)]^2, \tag{2.3}$$

with W the width of an image, H the height of an image, I_r the reference image, and I_e the image evaluated.

The PSNR is a ratio between the maximum value of a signal and the MSE :

$$P_{\text{SNR}} = 10 \cdot \log_{10} \left(\frac{max(I_r)^2}{MSE} \right), \tag{2.4}$$

with $max(I_r)$ the maximum possible pixel value of the image I_r.

Due to the three channels in YUV image, weight are applied on each channel of the image to determine the $P_{\text{SNR}_{YUV}}$:

$$P_{\text{SNR}_{YUV}} = \frac{6 \cdot P_{\text{SNR}_Y} + P_{\text{SNR}_U} + P_{\text{SNR}_V}}{8}, \tag{2.5}$$

with P_{SNR_Y}, P_{SNR_U}, and P_{SNR_V} the PSNR of the luminance component and the two chrominance components.

Another commonly used full reference quality metric is the Structural SIMilarity (SSIM) [11] which in comparison to the PSNR measure the structure similarity between two images rather than a pixel-wise difference. The SSIM is calculated on various windows of the image. The measure between two windows I_r and I_e of size $N \times N$ is :

$$SSIM(I_r, I_e) = \frac{(2\mu_{I_r}\mu_{I_e} + c_1) + (2\sigma_{I_r I_e} + c_2)}{(\mu_{I_r}^2 + \mu_{I_e}^2 + c_1)(\sigma_{I_r}^2 + \sigma_{I_e}^2 + c_2)}, \tag{2.6}$$

with μ_{I_r} and μ_{I_e} the average of I_r and I_e, $\sigma_{I_r I_e}$ the covariance of I_r and I_e, $\mu_{I_r}^2$ and $\mu_{I_e}^2$ the variance of I_r and I_e, c_1 and c_2 are two variables dependent of the dynamic range of a pixel, i.e., bit depth.

Other full reference methods using video information are also available, such as Video Multimethod Assessment Fusion (VMAF) [12] or MOtion-tuned Video Integrity Evaluation (MOVIE) [13] metrics. VMAF is based on Support Vector Machine (SVM) with four features as input to predict a single output score per video frame. The four features are the visual information fidelity, the detail loss metric, the mean collocated pixel difference and the anti-noise signal-to-noise ratio. A mean is performed on each frame score to provide a single value $\in [0, 100]$ for a video. MOVIE defines a spatial and temporal quality that are pooled together through the frame to obtain an overall video quality score. The spatial distortion is predicted with a space-time frequency decomposition of both reference and evaluated videos using a Gabor filter. The temporal distortion is calculated with motion information based also on the Gabor filter output.

2.2.1.2 Reduced and no reference methods

Reduced reference methods are used when the original video is not totally available. Different metrics are available for this type of problematic with for instance the SSIM Reduced-Reference (SRR) [14]. SRR is based on SSIM that extracts a score using a reference video pattern.

No reference method are developed to define the quality of a video without any reference. These metrics can be pixel-based or bitstream-based. Pixel-based metrics evaluate the degradation through specific types such as blurring or coding artefacts. Mittal *et al.* [15] proposed a blind image quality assessment based on multivariate Gaussian model with quality aware features derived from natural scene statistic. Bitstream-based metrics extract features from the bitstream such as motion vectors or quantization parameters. Saad *et al.* [16] determined a quality metric that is non-distortion specific. This approach predicts the video quality relying on a spatio-temporal model in the Discrete Cosine Transform (DCT) domain and on the type of motion occurring in the scene.

2.2.2 Subjective quality metrics

Measuring subjective video quality is necessary because objective quality assessment algorithms such as PSNR have been shown to correlate poorly with subjective ratings. Subjective metrics are highly related to the users satisfaction even if it is more expensive to perform in terms of time and human resources. Compared with objective metrics, subjective video metrics are dependent of human feelings. Indeed, subjective metrics are based on the analysis made by a group of viewers. Their opinions on the proposed videos are recorded and averaged into the Mean Opinion Score (MOS) to evaluate the quality of each video sequence. Following steps to evaluate the subjective video quality are typically followed :

- Source selection with representative number of contents, characteristics, and several settings that should be evaluated.
- Defining test environment with dedicated room based on recommendations [17, 18].
- Select large number of viewers under certain restrictions to avoid potential bias.
- Carry out testing in the predefined test environment.
- Calculate and analyze the results.

2.2.3 Complexity reduction metric

Through all this thesis, a complexity reduction metric ΔET is applied. The complexity reduction is defined by the difference of encoding complexity ΔET between a reference and a particular encoding averaged at four Quantization Parameter (QP)s. ΔET is computed as follows

$$\Delta ET = \frac{1}{4} \sum_{QP_i \in \{22,27,32,37\}} \frac{T_R(QP_i) - T_C(QP_i)}{T_R(QP_i)}, \tag{2.7}$$

where T_R is the reference encoding time of the VVC software and T_C the encoding time of the modified VVC software.

2.3 Versatile Video Coding test model and Common Test Conditions

The metrics described in the previous section are used to determine the performance of an encoded video compared with its reference video. In order to have fair comparison with the same software and sequences between researchers, CTC are defined by Joint Video Expert Team (JVET).

All experiments conducted in this thesis are applied on the VTM with the test video sequences defined in the VVC CTC [19]. The VTM is the VVC reference software which includes all the tools of the standard. The CTC video sequences are separated in seven classes as follows : A1 (3840×2160), A2 (3840×2160), B (1920×1080), C (832×480), D (416×240), E (1280×720), and F (832×480 to 1920×1080). These classes feature different frame rates, bit depths, motions, textures, and spatial resolutions. Class A to E are natural sequences and class F contains specific screen content sequences. These video sequences are encoded at four QP values : 22, 27, 32, and 37. The TABLE 2.1 summarizes the CTC sequences with their respective width, height, and bit depth.

Table 2.1 – *CTC sequences with their respective width, height, and bit depth.*

Class	Sequence	Width	Height	Bit depth
Class A1	Tango2	3840	2160	10
	FoodMarket4	3840	2160	10
	Campfire	3840	2160	10
Class A2	CatRobot1	3840	2160	10
	DaylightRoad2	3840	2160	10
	ParkRunning3	3840	2160	10
Class B	MarketPlace	1920	1080	10
	RitualDance	1920	1080	10
	Cactus	1920	1080	8
	BasketballDrive	1920	1080	8
	BQTerrace	1920	1080	8
Class C	RaceHorses	832	480	8
	BQMall	832	480	8
	PartyScene	832	480	8
	BasketballDrill	832	480	8
Class D	RaceHorses	416	240	8
	BQSquare	416	240	8
	BlowingBubbles	416	240	8
	BasketballPass	416	240	8
Class E	FourPeople	1280	720	8
	Johnny	1280	720	8
	KristenAndSara	1280	720	8
Class F	ArenaOfValor	1920	1080	8
	BasketballDrillText	832	480	8
	SlideEditing	1280	720	8
	SlideShow	1280	720	8

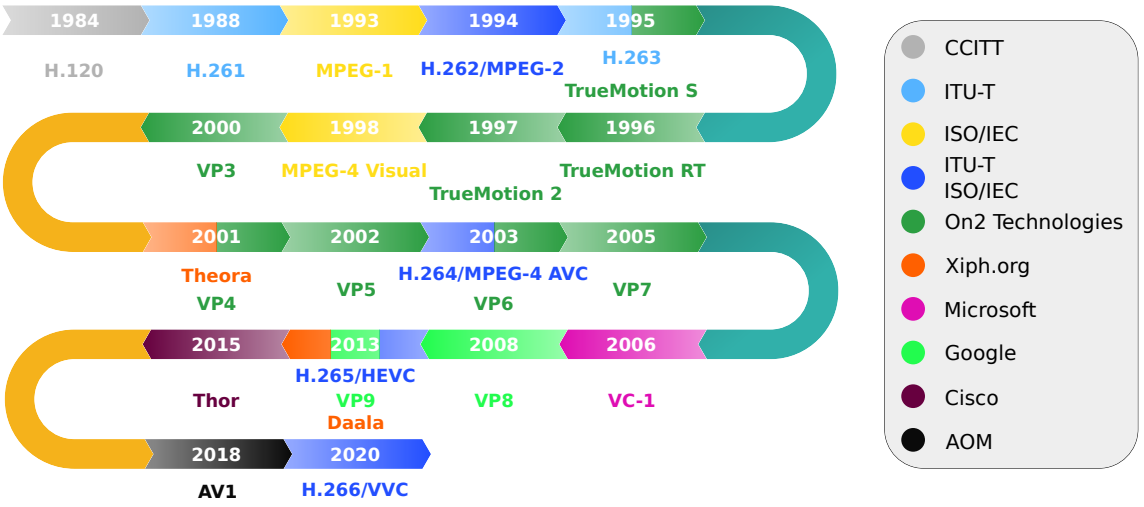


Figure 2.5 – *Video coding standard publication through the years.*

2.4 Video coding standards

Video coding is a method that compresses the data of a video by maximizing its compression while minimizing the impact on the video quality through correlations between pixels, these correlations can be both spatial and temporal. The video coding is related to an encoder that transforms a video into a bitstream and a decoder that transforms a bitstream into a video. A video coding standard describes a bitstream structure and the decoding process of this bitstream in order to reconstruct the video. Indeed, the standard is not defining the encoding process. Encoders can have different implementations by selecting essential tools according to the desired coding and complexity performance. The only requirement for encoder is to produce compliant bitstream of the standard. However, decoders must implement all the tools defined in the standard to be compliant.

The standardization process follows several steps. Firstly, requirements are identified for new applications or media such as 8K, high dynamic range, or 360-degree videos. Then, a call for evidence demonstrates that a significant gain over the most recent standard has been achieved with different algorithms proposed by different contributors that are going to be the core of the standard. As a consequence, a call for proposal begin with tools proposition to enhance the quality of the standard. Finally the standard is published internationally.

Figure 2.5 presents different video coding standards published from 1984 until nowadays. The first digital video coding standard has been published in 1984 as H.120 by the International Telegraph and Telephone Consultative Committee (CCITT). H.120 is the only standard based on Differential Pulse-Code Modulation (DPCM), its performance is not sufficient to be used in practice. H.261 is the first practical DCT-based standard. Since H.261, all major video coding standards have adopted the DCT as a fundamental operation in their process. International Organization for Standardization (ISO), International Electrotechnical Commission (IEC) and International Telecommunication Union Telecommunication Standardization Sector (ITU-T) (previously CCITT) are the main publisher of standard with On2 Technologies. The three historical institutions ISO, IEC, and ITU-T join forces to create JVET and proposed H.262/MPEG-2 Part 2, H.264/MPEG-4 Advanced Video Coding (AVC), H.265/HEVC, and VVC ITU-T H.266 | MPEG-I - Part 3 (ISO/IEC 23090-3).

In 2019, Bitmovin proposes a survey [20] that studies the global share of video codecs. 91% and 43% of companies used in 2019 AVC and HEVC, respectively. These standards are and will probably be the most widely used in the coming years with the newly standardized VVC. In order to compete with those standards developed by JVET, Amazon, Cisco, Google, Intel, Microsoft, Mozilla, and Netflix created a non-profit industry consortium Alliance for Open Media (AOM) that standardized AOMedia Video 1 (AV1) in 2018 based on VP9, Daala, and Thor. AV1 aims to be royalty free with the same or better coding efficiency compared with JVET standards.

Kerdranvat *et al.* [21] proposed a comparison between HEVC, AV1, and VVC through objective quality metric and processing time. They established that for UHD sequences, VVC outperforms HEVC with almost 42% BD-BR gain for the same PSNR. AV1 has a gain of 18% in terms of BD-BR compared with HEVC which is lower than the BD-BR gain of VVC compared with HEVC. The gains are homogeneous when the BD-BR gains are computed based on the same VMAF or Multi-Scale Structural Similarity (MS-SSIM) values compared with the same PSNR. In terms of processing time, VVC is the most complex with 1308% of encoding runtime and 192% of decoding runtime compared with HEVC. AV1 is less complex than VVC. Indeed, when compared with HEVC, AV1 reaches 497% and 76% of encoding and decoding time, respectively. However, the VVC software is only used for the standardization while the AV1 software is suited for production. This leads to software optimizations for AV1 through its 3 years of existence with improved running time for the same quality. The fact that VVC is more complex than AV1 can be explained partially by these optimizations.

2.5 Versatile Video Coding

VVC is the most recent video coding standard following the classical hybrid video coding scheme which combines both intra and inter prediction plus the transform coding. The global presentation of an encoder proposed in this section focuses on the VVC standard which is the standard considered throughout this thesis. Yet, these descriptions outline general principle of video coding that can be extended on different video coding standards.

Figure 2.6 presents the VVC encoder block diagram with the main steps of encoding. It takes as input a video and outputs a bitstream that represents this video in a compressed way with a sequence of bits. This bitstream is then processed by a decoder that recreates the video with quality degradation. During the generation of the bitstream, several tools are performed through the encoding part in order to maximize the decoded video quality and minimize the size of the bitstream.

A global presentation of these tools are proposed as follows :

- Tree partitioning : divides a frame into blocks based on the feature of these blocks. Large blocks are most likely a texture with low variance, instead, small blocks are the part of the frame with a lot of details.

- Prediction : this process aims at predicting a block of pixels from other pixels. In the case of intra prediction, the considered block of pixels is predicted from its neighbours in the same frame in order to exploit the spatial redundancy. In the case of inter prediction, the considered block of pixels is predicted from a block of pixel located in another frame in order to exploit the temporal redundancy. The predicted block is an approximation of the considered block. This predicted block is then subtracted to the considered block in order to create the residual block. This residual block contains the residues which are processed in the following part of the encoder.

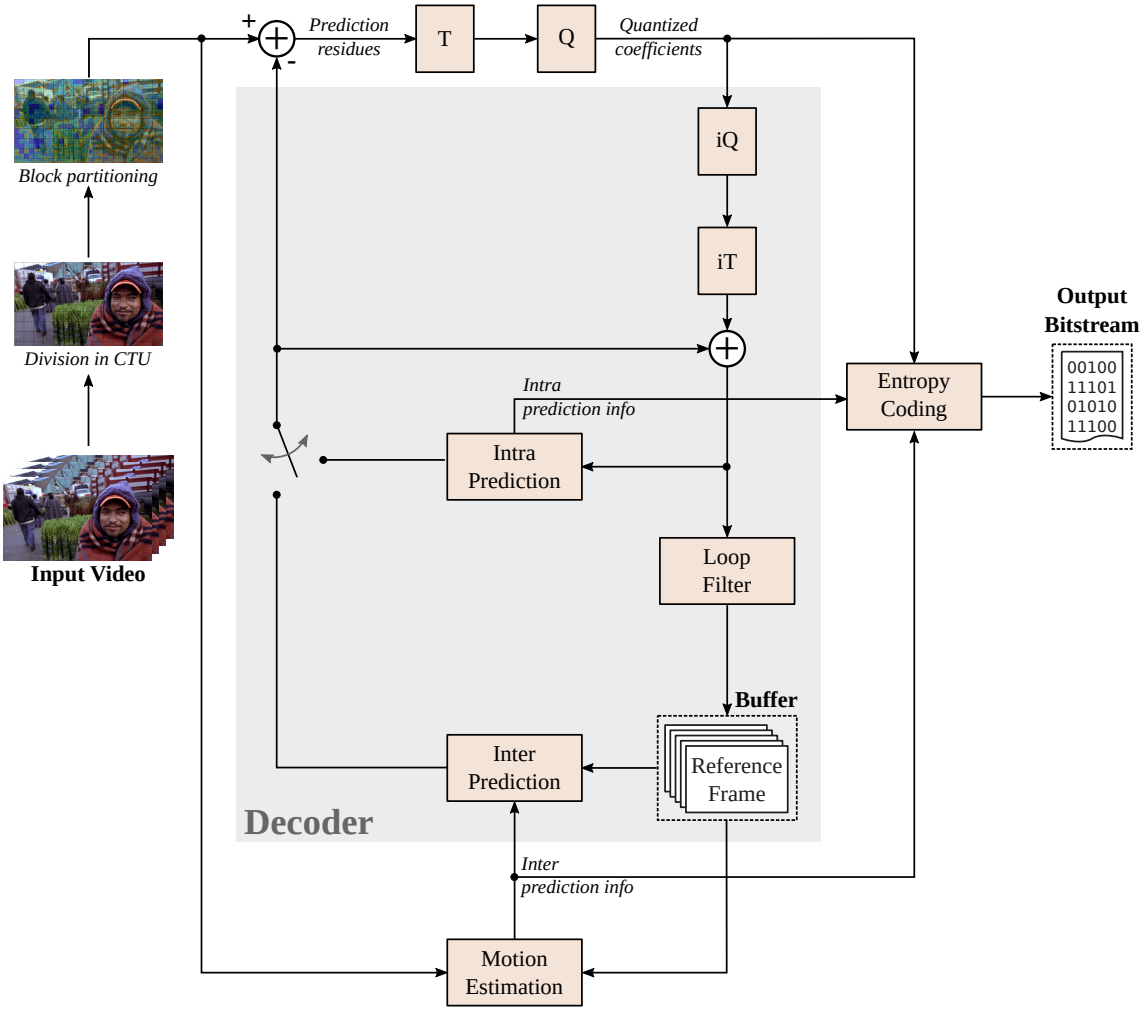


FIGURE 2.6 – *VVC block diagram.*

- Transform : takes as input the residues and output these residues into the frequency domain. Transform and particularly DCT and Discrete Sine Transform (DST) have a strong energy compaction property. This tends to concentrate most of the signal information in a few low-frequency components.

- Quantization : maps the frequency information that is dispatch on a range of values into a smaller range of values. This is the only encoding process that leads to information losses, indeed, it is an irreversible process.

- Decoder : is processed inside the encoder to enable the prediction. For intra prediction, the inverse quantification and inverse transform are processed, then the predicted block of the reference block is added to the output of the inverse transform. These succession of process results in the reconstructed block. For inter prediction, the same process is applied supplemented by the in-loop filters (Luma Mapping with Chroma Scaling (LMCS), Deblocking Filter (DBF), Sample Adaptive Offset (SAO) and Adaptive Loop Filter (ALF)) which creates the decoded block.

- Context-Adaptive Binary Arithmetic Coding (CABAC) : is the last encoding step, it is based on arithmetic coding, a form of entropy coding. It takes the quantized coefficients plus several information such as intra or inter prediction in order to produce the bitstream.

2.6 Encoding tools

This section explains in details the different processes included in the VVC encoder and their features.

2.6.1 Tree partitioning

The first encoding process of VVC is the tree partitioning that optimizes the division of the frame into blocks. These blocks are the support for the all encoding process starting by the prediction.

2.6.1.1 Coding Tree Unit

Figure 2.7 presents a frame divided by N_{CTU} which is calculated with $N_{CTU.W}$ and $N_{CTU.H}$. $N_{CTU.W}$ defines the number of Coding Tree Unit (CTU) in width and $N_{CTU.H}$ is the number of CTU in height. A CTU consists of a block of luminance and the two corresponding blocks of chrominance. It is defined by its width W_{CTU} and height H_{CTU}. In the VTM, W_{CTU} and H_{CTU} are specified to be 128×128 pixels such as the maximum allowed size of the luminance block.

The encoder starts by dividing the frame into CTUs. Then, the tree partitioning initiates its process with the CTUs. Such as presented in Figure 2.7, the raster-scan which determines the order to process the CTUs during the block partitioning starts with the top left of the frame to the bottom right.

2.6.1.2 Slice and tile

Picture can be divided into tiles and slices. Tiles and slices are rectangular regions of the frame.

Figure 2.7 – *Picture divided in CTUs.*

Tiles have been proposed to increase the capability of parallel processing. The separation between tiles are defined by straight lines that cross the whole frame. The tiles can be processed in parallel which means that the encoding time is faster based on the number of tiles and threads but the dependencies are broken between the tiles. These broken dependencies and the CABAC reinitialization at each tile lead to coding efficiency loss. A tile is independently decodable with some shared header information at encoding.

Slices have independent syntax elements that can be parsed from the bitstream and that can be correctly decoded without any other slice information. Indeed, each slice is integrated into one Network Abstraction Layer (NAL) which is interesting to make data available for a system with fragmented packetization at low latency. Due to the independence between the slices, slice can be encoded and decoded in parallel such as the tiles. However, slice can be created through two different slice modes. Indeed, the raster-scan slice contains complete and consecutive tiles in the raster-scan order of the frame. The second mode is rectangular slice that divides a tile with horizontal separation.

Figure 2.8 presents a frame separated into tiles in blue and slices in pink. Raster-scan slice are represented in slice number 2 which contains tiles B, C, D, and E. Three rectangular slices are shown with slice 1, 3, and 4.

2.6.1.3 Coding Unit

The new tree partitioning scheme proposed in VVC is the most efficient coding tool integrated in the standard [22]. The tree partitioning starts the process on a root block named CTU, i.e., a block of 128×128 pixels in the VTM. The blocks resulting from the tree partitioning process are named Coding Unit (CU)s and may have a size between the CTU size and 4×4 pixels. Figure 2.9 presents the splits supported by VVC. Quad-Tree (QT) divides a CU into four equal sub-CUs. Moreover, VVC allows rectangular shape for CU with its novel splits Binary-Tree (BT) and Ternary-Tree (TT). The BT divides a CU in

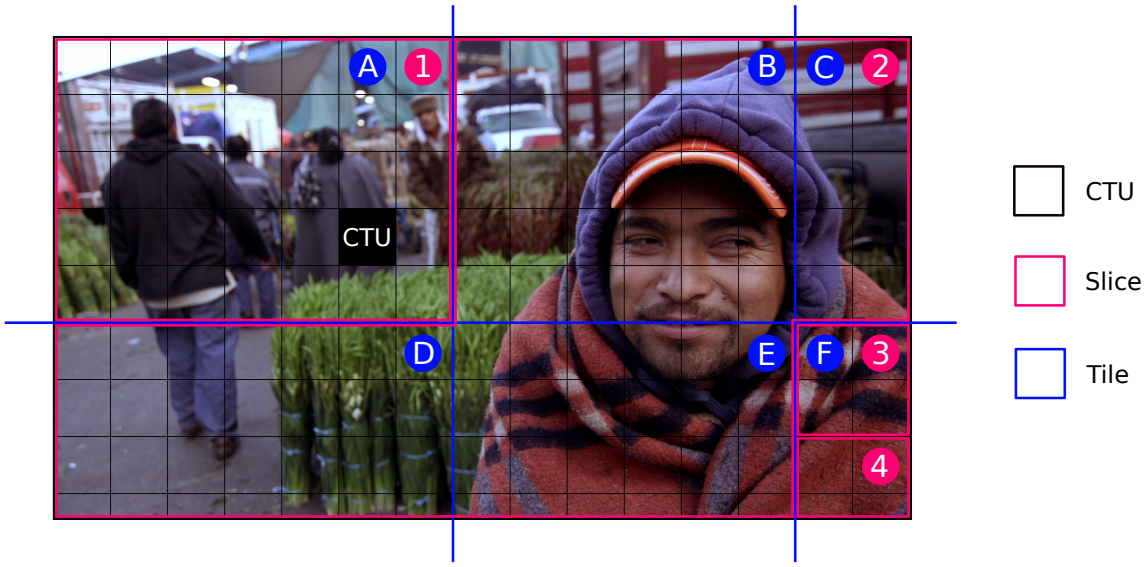


Figure 2.8 – *Example of slices and tiles in a frame.*

two sub-CUs while the TT divides a CU in three sub-CUs with the ratio 1 :2 :1. Both BT and TT can split a CU horizontally or vertically.

Figure 2.10 presents a frame in the background and its luminance partition in the foreground performed by the VTM at QP 27. Several colors are display on the CUs depending on their depth, from dark blue which are large CUs to red which are small CUs. The CU sizes are defined relying on their block characteristics. Indeed, CTUs containing flat texture such as the bottom left of the frame results in large CUs. However, CTUs with a lot of details such as the CTU surrounded by a red box near the eye results in small CUs.

Some restrictions [8] are applied on the VTM to avoid that different splitting patterns result in the same partition structure or the examination of irrelevant partition :

- QT when BT or TT is already performed once.
- QT, BT, or TT when QT, BT, or TT maximum depth is exceed or CU size restrictions are exceeded.
- BT for the central partition of a TT in the same direction.
- QT when BTs were evaluated and did not reduce the Rate-Distortion (RD)-cost.
- QT enforced or prohibited based on neighbouring blocks.
- BT or TT at predefined depths when skip mode has the best RD-cost of non-split mode.
- QT, BT, or TT when the resulting RD-cost of those splits are superior of a threshold based on the non-split RD-cost.
- TT when non-split has no residual and BT in the same direction did not reduce the RD-cost.

A dual tree tool is proposed in VVC for intra frame. A separate partition tree can be used for the chrominance components in order to better take into consideration the chrominance characteristics. The same process as the one used for luminance is carried-out

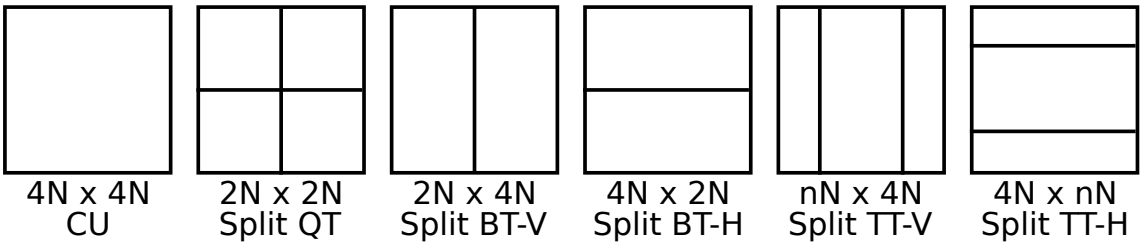


Figure 2.9 – *Available split included in VTM for a $4N \times 4N$ CU.*

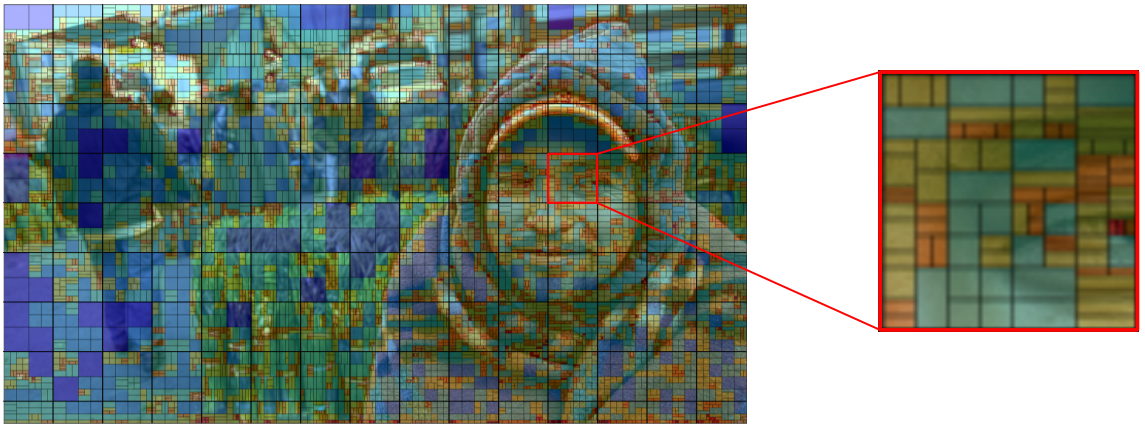


Figure 2.10 – *Example of a frame partitioning and zoom in a specific CTU.*

for the chrominance samples. It is interesting to note that to reduce the buffer requirement in the VTM, chrominance partitioning under the $4:2:0$ format starts at a CTU size of 64×64 with a first implicit QT.

Picture boundaries have specific partition process when a CTU exceeds the bottom or the right of the frame, i.e., specific partition are computed when the picture size is not a multiple of the CTU size. These blocks that exceed the frame need are pre-processed before the tree partitioning process starts. This preprocess consists of implicit splits performed while the block size is not fully inside the frame. Only QT and BT are applicable for implicit splits depending on the splitting rules defined by VVC.

2.6.2 Intra mode

Intra prediction mode exploits the spatial redundancies by predicting the content of a CU through a sample of an already encoded CU in the same frame. The already encoded CU is a spatial neighbour of the computed CU, i.e., left or above CU of the computed CU. Once the prediction block is defined, the difference is made between the reference CU and its prediction in order to obtain the residues.

Several intra prediction tools has been introduced in VVC. Indeed, angular mode has been extended to 65 plus DC and planar mode. Figure 2.11 presents a CU and its two neighbouring samples of CUs previously encoded in red (top CU) and blue (left CU). Based on these samples, an angular mode can be selected to project the reconstructed sample on the current CU. Non-square blocks bring a new issue as the conventional angular intra prediction directions are comprised between 45 and -135 degrees in clockwise direction. Wide-angle intra prediction such as presented in Figure 2.11 resolves the non-square block issue by replacing conventional angular modes to wide-angle mode based on the ratio W/H.

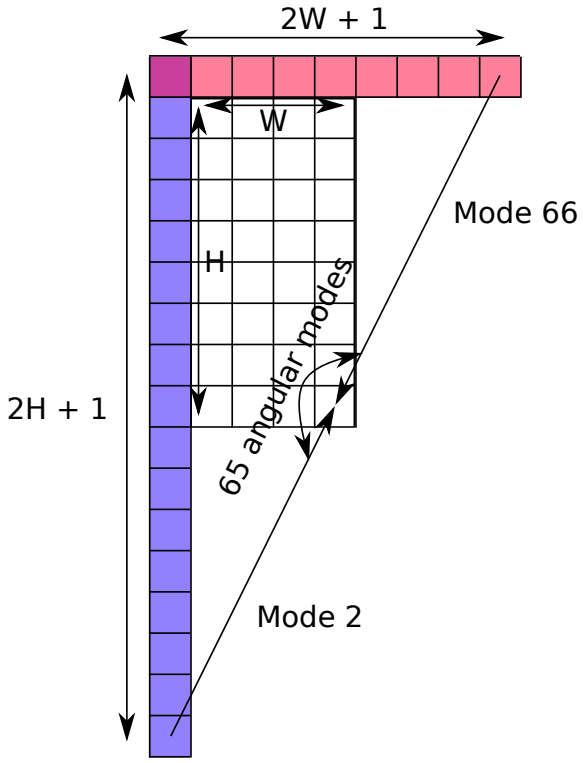


Figure 2.11 – *Wide-angle intra angular mode with non-square block.*

Other tools are proposed such as the four-tap interpolation filters which improve the directional intra prediction accuracy, the cross-component linear model which predicts the chroma samples based on the luminance samples of the same CU, the Matrix-Based Intra Prediction (MIP), the position dependent intra prediction combination, the Multiple Reference Line (MRL), and the Intra Sub-Partitions (ISP). The MRL proposes the utilization of two additional lines of the above CU instead of one and two additional columns of the left CU instead of one. These three lines and columns can be used as the reconstructed samples for the intra mode prediction. Finally, the ISP further split the CU horizontally or vertically into two or four sub-partitions depending on the block size. With CU of size 4×8 or 8×4, the tool proposed two sub-partitions. For larger CU size, it proposed four sub-partitions.

2.6.3 Inter mode

Unlike intra mode, inter mode exploits the temporal redundancies by predicting the content of a CU by using CU of previously encoded frame. VVC specified new refined inter prediction coding tools beyond HEVC inter prediction tools which are listed as follows :

- Extended merge prediction : the merge candidate list is constructed in VVC with the spatial Motion Vector Prediction (MVP) from spatial neighbour CUs, temporal MVP from collocated CUs, history-based MVP from a First In First Out (FIFO) table, pairwise average MVP and zero MVs.

- High precision motion compensation and motion vector storage : VVC increases the MV precision to 1/16 luminance sample which is particularly interesting for affine mode.

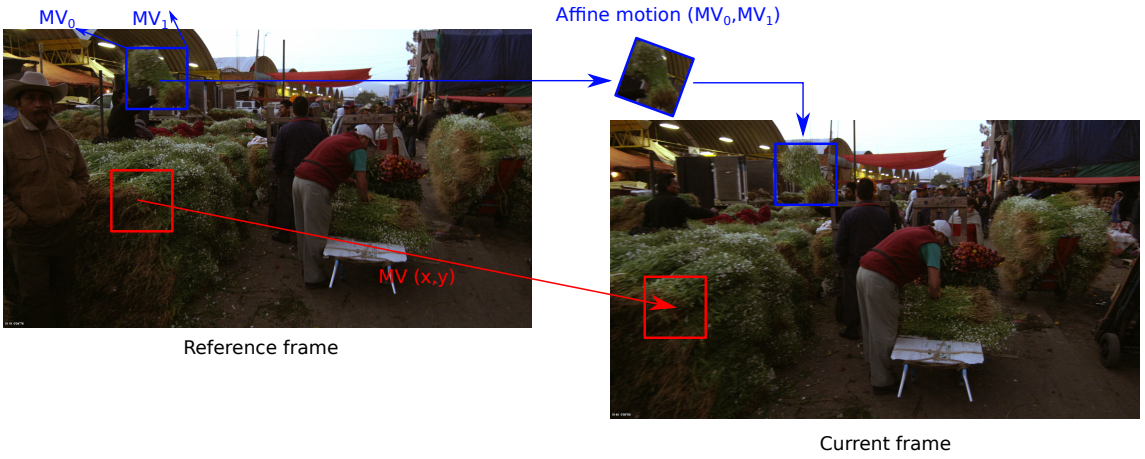


Figure 2.12 – *Motion Vector (MV) between the reference frame and the current frame.*

- Merge mode with Motion Vector Differences (MVD) : a merge further refined by a merge candidate flag which allows two possible merge candidates, an index for motion magnitude, and another index to specify motion direction.
- Symmetric MVD coding : for bi-prediction MVD, the MVD1 is set to -MVD0 to avoid signaling MVD1 and the indices of the reference pictures.
- Affine motion compensated prediction : 2 or 3 MVs which allow to zoom in-out, rotate, generate perspective, or create other irregular motions that appear in video sequences.
- Sub-block based temporal motion vector prediction : is the same as temporal motion vector prediction tool in HEVC with motion at sub-CU level instead of CU-level and it applies a motion shift before fetching the temporal motion information.
- Adaptive motion vector resolution : allows MVD of the CU to have different precision from 1/16 to 4 luminance sample, different range are proposed based on normal or affine adaptive motion vector prediction mode.
- Bi-prediction with CU-level weight : extend the bi-prediction with weighted average with five weights applicable instead of simple average.
- Bi-directional optical flow : refines the bi-prediction signal of a CU by calculating a motion refinement for each $4x4$ sub-CU.
- Decoder side motion vector refinement : derives a refined MV which generates the inter prediction samples and which is also used in the temporal motion vector prediction for future pictures coding.
- Geometric partitioning mode : another split can be performed by a geometrically straight line defined with an angle and an offset up to 64 different partitions.
- Combined inter and intra prediction : combines with weighted averaging the regular merge mode for inter prediction and the planar mode for intra prediction.
- Reference Picture Resampling (RPR) : enables resampling in the motion compensation process by processing together the scaling ratio with the motion information

to locate the reference samples. It is performed at a block level instead of a new sequence parameter set. In VVC, the scaling ratio from the reference picture to the current picture is restricted between 2 times downsampling to 8 times upsampling.

- Miscellaneous inter prediction aspects : reduces the memory bandwidth by disabling inter 4×4 CU. For 4×8 or 8×4, only unidirectional mode is proposed and finally when merge mode is bidirectional, only the motion information from the list 0 is kept.

Figure 2.12 presents the current encoded frame and one of its reference frame. The red CU in the current frame takes as best inter mode the MV with two parameters (x, y). These parameters, x and y, define the movement between the reference frame and the current frame of the most related CU compared with the current CU through the two horizontal and vertical axes. Another tool is illustrated by blue CUs with the affine motion compensated prediction. Unlike the MV with two values which defines the movement, the affine motion provides two or three MVs to define the movement. Affine motion compensated prediction tool has been introduced in VVC to match with more kind of motion than simple MV. It allows the rotation, zoom in and out, perspective, and other irregular motions. For instance, the blue CU of the reference frame rotates in a clockwise direction to better match the current CU.

2.6.4 Transform

The aim of the transform process is to compact the energy of the residues that are created by intra or inter prediction. The transform process translates the residues into the frequency domain and allows concentrating the signal information in low-frequency components. Figure 2.13 presents the transformation of an image representing an eye with DCT-II performed on the whole 64×64 block. Several transform are available for the luminance part with the newly introduced Multiple Transform Selection (MTS), i.e., DCT-II, DCT-VIII, and DST-VII. The kernels (basis functions) of these transforms are computed as follows :

$$DCT2 => T_i(j) = \omega_0 \cdot \sqrt{\frac{2}{N}} \cdot \cos\left(\frac{\pi \cdot i \cdot (2j+1)}{2N}\right), \tag{2.8}$$

$$DCT8 => T_i(j) = \sqrt{\frac{4}{2N+1}} \cdot \cos\left(\frac{\pi \cdot (2i+1) \cdot (2j+1)}{4N+2}\right), \tag{2.9}$$

$$DST7 => T_i(j) = \sqrt{\frac{4}{2N+1}} \cdot \sin\left(\frac{\pi \cdot (2i+1) \cdot (j+1)}{2N+1}\right), \tag{2.10}$$

where $\omega_0 = \begin{cases} \sqrt{\frac{2}{N}} & i = 0 \\ 1 & i \neq 0 \end{cases}$ and N is the N-point input. These new transforms forces VVC to create the MTS CU flag. If this flag is disabled, only the DCT-II is performed for both horizontal and vertical transform. However, when the flag is up to one, an horizontal and vertical flag is additionally signalled to indicate if DST-VII or DCT-VIII has been performed horizontally or vertically.

In VVC, the transform can be applied on large block-size, up to 64×64, which is particularly interesting for high resolution sequences. To reduce the complexity of large block with width or height equal to 64, the high frequency coefficients are set to zero, i.e., with width equal to 64 only the left 32 columns are kept and for height equal to 64 only

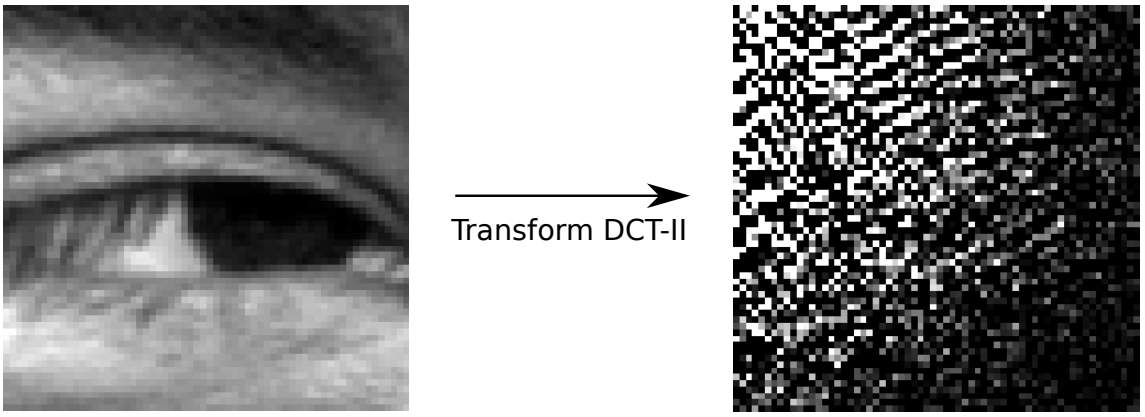


Figure 2.13 – *Application of DCT-II on an image representing an eye.*

the top 32 rows are kept. For DST-VII and DCT-VIII, the high frequency coefficients are also set to zero for blocks with width or height equal to 32.

Low-Frequency Non-Separable Transform (LFNST) [23] is a tool adopted in VVC that perform a matrix multiplication to further compress the redundancy between low-frequency primary transform coefficients. The matrix multiplication is applied after the forward primary transform and before quantization at encoding such as :

$$\vec{F} = T \cdot \vec{X}, \tag{2.11}$$

with $\vec{F}$ the resulted transform coefficient vector, T the transform kernel, and $\vec{X}$ the result of the forward primary transform. Four transform sets are available, with each containing two transforms. The transform set selection is determined from the intra prediction mode.

For inter-predicted CU, the sub-block transform is proposed. With this mode, only a sub-part of the residual block is transformed and the other part is set to zero. The sub-block transform type and position information are transmitted in the bitstream. Block can be split horizontally or vertically in two parts with ratio 2 :2 such as a binary split or 1 :3/3 :1 such as an asymmetric binary split. The transform sub-block can be selected when a block is divided with binary split, unlike asymmetric binary split which restricts the small sub-block to contain the non-zero residual. The MTS is applied on the non-zero residual sub-block depending on the sub-block transform type and position except for a sub-block larger than 32 in width or height which implies the DCT-II transform.

2.6.5 Quantization

Quantization maps input values from a defined set into output values in a smaller set. The aim of quantization is to decrease the bitrate while maintaining low quality loss in reconstruction. Quantization in video coding takes as input the transform coefficients that are concentrated into a small range of coefficients which is more effective than the original block. VVC quantizer design is based on scalar quantization with uniform reconstruction quantizers in which the set of admissible reconstruction values is specified by the quantization step. Two other quantizers are included in VVC with sign data hiding that omit the coding of the sign and derives it from the parity of the sum of absolute values of quantization index and trellis-coded quantization which obtains several quantization candidates based on the pre-quantization results given a transform coefficient. Depending on the application requirements, the encoder selects a quantizer design which is indicated in

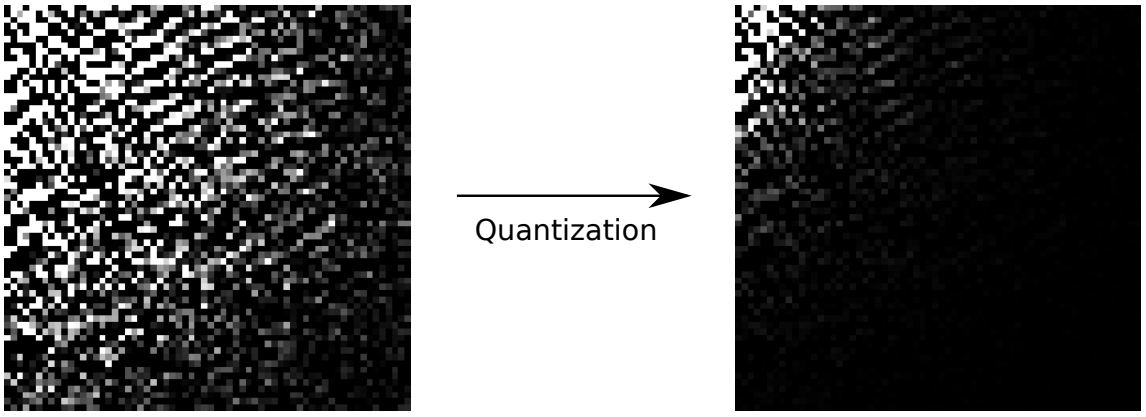


Figure 2.14 – *Application of quantization on transformed image representing an eye.*

the slice header. Figure 2.14 presents the transformed coefficients of the previous image representing an eye quantized by a standard quantization table.

This process leads to loss of information guided by a parameter named QP. Indeed, the quantization step is defined with an exponential relationship to the QP. QP is ranging between 0 and 63 and handles the size of quantization step. Increasing the QP induces more information losses in the transformed coefficient.

2.6.6 In-loop filters

Three in-loop filters are processed in the following order on the reconstructed samples in VVC with deblocking filter [24], SAO [25], and ALF [26]. These filtering techniques improve the visual quality and the coding efficiency of the sequence.

Before in-loop filters application, VVC introduced the LMCS [27] which contains the luma mapping and luma-dependent chroma residue scaling. LMCS improved the coding efficiency by adaptively modifying the coded samples distribution. The luma mapping redistributes the codewords of the luma input signal within the complete codeword range. The chroma residual scaling needs the luma mapping signalled to be available. Chroma scaling aims to compensate the impact of luma remapping to its corresponding chroma signal by applying a constant scaling factor to all chroma residues.

The first in-loop filter processed is the deblocking filter which reduces the artefacts at block boundaries created by discontinuities between neighbour blocks due to their independent coding. VVC uses the same process as HEVC for the deblocking filter. The filter is applied on CU boundaries plus in prediction sub-block boundaries and transform sub-block boundaries which are introduced by sub-block based temporal motion vector prediction or affine mode and by sub-block transform or intra sub-partitions, respectively. Its processing order starts with horizontal filtering for vertical edges followed by vertical filtering for horizontal edges with each performed on the whole frame. This order enables parallel processing on several horizontal or vertical filtering.

The second filter performed is the SAO which classifies the reconstructed pixels into different categories and then reduces the distortion by adding an offset based on the category specified. This filtering reduces the ringing artefacts due to the quantization.

Finally, ALF is separated in three independent process with luma ALF, chroma ALF and cross component ALF in order to enhance the reconstructed signal. Several filter are enabled to be applied in a block-based way for each 4×4 block. The dimension of luma and chroma ALF are a 7×7 and 5×5 diamond shape, respectively. For luma ALF, a

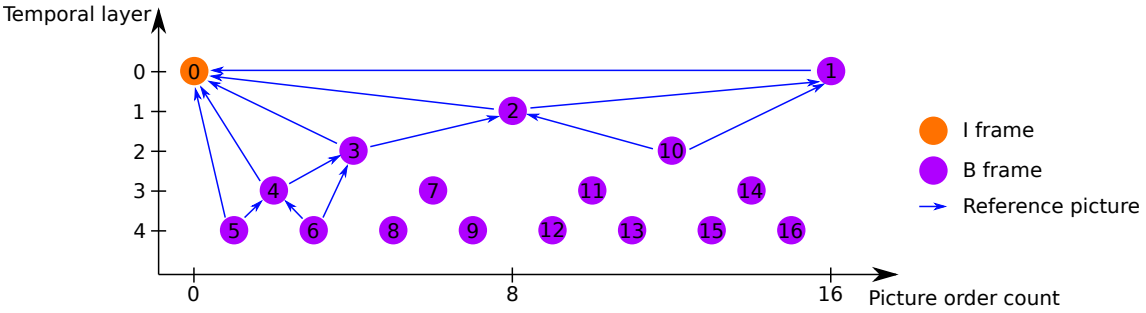


Figure 2.15 – *Temporal relationship between frames on RA configuration.*

classification method is applied to select one among the 25 filters based on the direction and activity of local gradients of each 4×4 block. Geometric transformations such as rotation or diagonal and vertical flipping are enabled for the filter coefficients and the clipping values based on the 4×4 luma block gradients.

2.6.7 Entropy encoding

The Context-Adaptive Binary Arithmetic Coding is the entropy encoding of VVC for all low-level syntax elements. It adapts the number of bits used to describe an element based on the appearance probability of this element (the context), i.e., frequent elements are represented by a few bits while uncommon elements are represented with more bits. Syntax elements includes all the information needed to reconstruct the video at the decoder side. When syntax elements are non-binary, they are mapped to binary codewords. All the binary symbols with the binary syntax elements and the codewords are coded through the CABAC. The CABAC supports two operating modes, a regular mode, and a bypass mode. The bypass mode bypasses the whole binary arithmetic encoding when elements are assumed as uniformly distributed. In regular mode, the binary symbols are coded with binary arithmetic coding where the probability model is selected either by, the type of syntax element and the bin position or index of the binarized syntax element, or, related information such as the size of CU.

2.6.8 Encoder configurations

The VTM is proposed with three different configurations All Intra (AI), Random Access (RA), and Low Delay (LD). These configurations differentiate each other by the possibility offered to encode a frame and especially the prediction. Indeed, inter prediction is not always available depending on the encoding configuration or the type of frame computed. A frame is defined by a type which is an I frame, a P frame, or a B frame. I frame is only depending on itself, no temporal relationship are applicable during the encoding process, i.e., inter prediction are not available on I frame as no frame has already been encoded. In contrast, P and B frames can use already decoded frames to increase the encoding efficiency. P frame uses one reference frame for inter prediction, however, B frame uses two reference frames for inter prediction with tools such as bi-prediction.

In AI configuration, no temporal relationship between the frames are allowed, only intra prediction is available. Each frame of the sequence is spatially encoded as an I frame and one QP is specified for the whole sequence. The frames are encoded in the same order than their temporal location in the sequence.

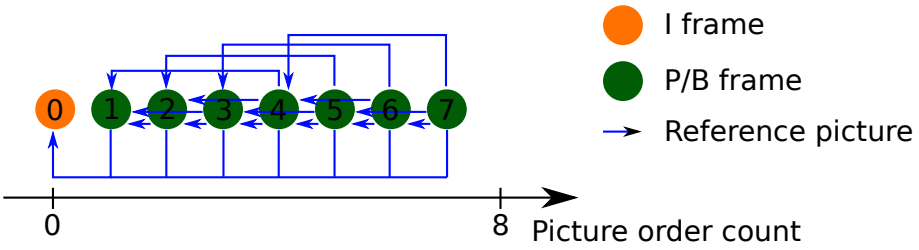


Figure 2.16 – *Temporal relationship between frames in LD configuration.*

In RA configuration, temporal relationship are allowed between frames by a hierarchical B structure, i.e., the prediction can be computed with inter prediction except for the I frames inside the sequence. However, the intra prediction is also available for this configuration. Figure 2.15 presents the frames and their reference frames based on their temporal layer which is the hierarchical level of a frame. The reference frames are the frames that the computed frame uses for inter prediction, only a part of all the reference frames are shown in this figure for simplicity. The Picture Order Count (POC) which is the display order is also presented in comparison with the picture encoding order. In RA configuration, the first frame to be encoded is always an I frame, other frames included in the Group of Pictures (GOP) are B frames. The QP is picture dependent and is derived based on the temporal layer.

The LD configuration is proposed to evaluate the low-delay coding performance with LD-P or LD-B. These two specifications determines the type of frames of the GOP except the first one which is an I frame. In LD configuration, the POC is the same as the encoding frame order. Figure 2.16 presents the LD configuration through a GOP of 8 frames and their reference frames. Each P or B frames have a maximum of four frames that are able to be used as reference. The QP is also fluctuating in this configuration based on the location of the frame in the GOP.

2.7 Rate-Distortion Optimization

The RDO is a process that optimizes the different parameters proposed in VVC through the RD-cost. It minimizes the RD-cost which is defined as :

$$J = D + \lambda R, \tag{2.12}$$

with D the distortion between the original video and the encoded video, λ the Lagrangian multiplier, and R the rate. The RDO calculates the RD-cost recursively for all the CUs, i.e., the CUs created by the tree partitioning process with several prediction mode and transform tested. These RD-costs are then compared with select the minimum RD-cost with the optimal parameters for each CU. This optimization can be represented as :

$$\{p*, m*, t*\} = \underset{partition}{\arg\min}(\underset{mode}{\arg\min}(\underset{transform}{\arg\min}\ J(B_r, B_c(p, m, t)))), \tag{2.13}$$

with J the RD-cost, B_r the reference block, B_c the candidate block composed of p its partitioning, m its prediction mode, and t its transform. The optimization results in a selection of parameters p, m, and t that minimizes the most the RD.

Figure 2.17 is the representation of Eq. 2.13. Each step of the RDO is presented in this figure starting with the block partitioning, then the intra and inter mode prediction and finally the MTS. The first step, i.e., the block partitioning, optimizes the size of each

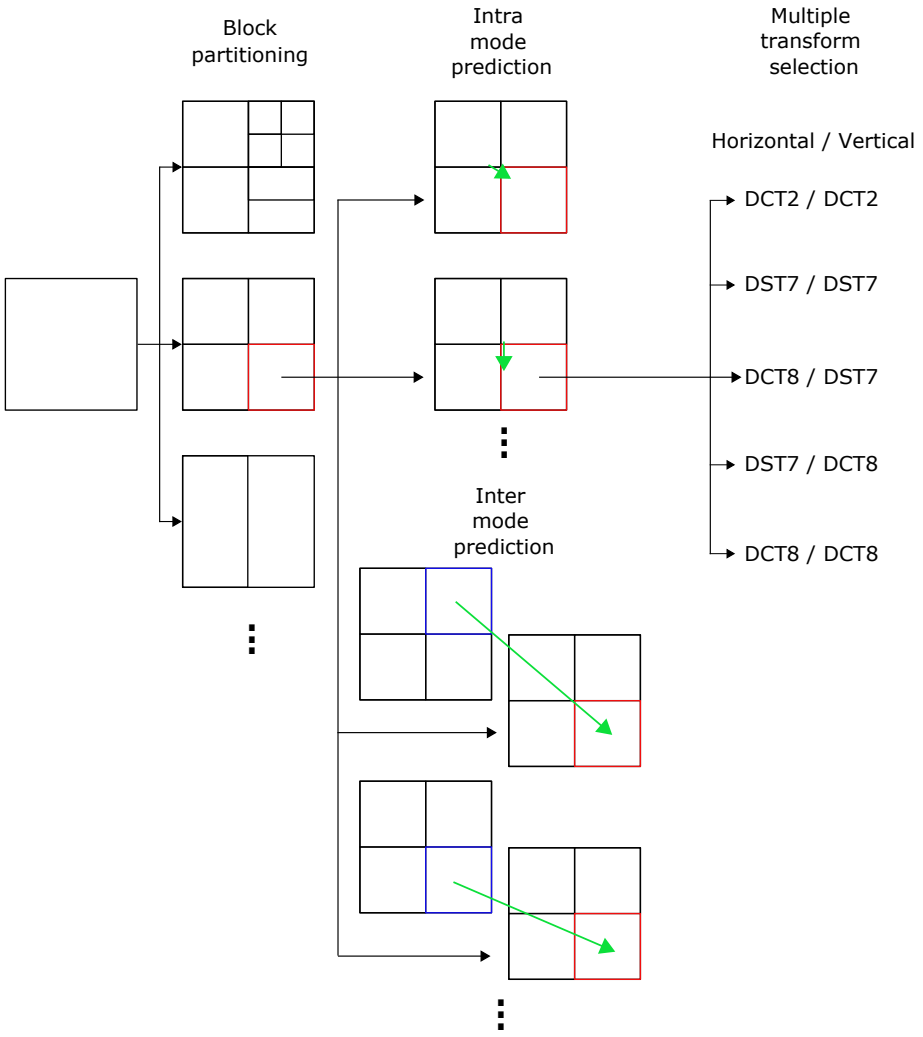


Figure 2.17 – *Rate-Distortion Optimization search process.*

CU depending on their characteristics. The second step is the prediction through intra or inter mode, which selects the more appropriate reference block for each CU tested in the tree partitioning process. The third and final steps decide the suitable horizontal and vertical transform among the MTS. This RDO results in a high complexity through all the RD-cost calculated by proceeding all the tools for each available configuration. Indeed, each CU proposed by the tree partitioning are predicted with each available prediction mode and then, the residues created are transformed with the MTS that has also several possibilities.

2.8 Progression of Versatile Video Coding test model coding efficiency and complexity

VVC has progress in terms of encoding quality through the standardization with the integration of new tools. As presented earlier, the VTM is the VVC reference software that implements all the tools included in the standard. This VTM software allows a fair comparison between the proposed tools during the standardization. The different proposals have been developed on several versions of the VTM to get comparative results. The incorpo-

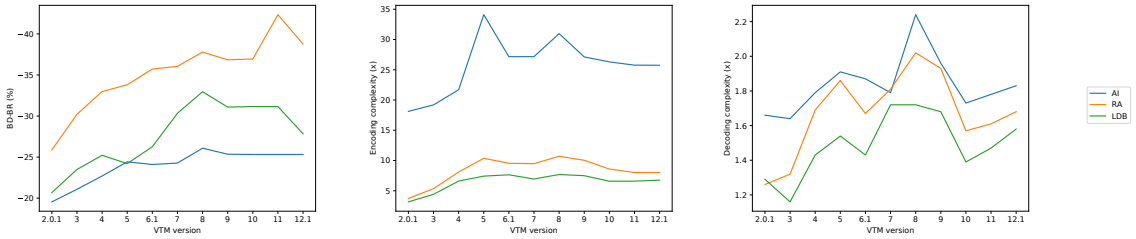


Figure 2.18 – *Comparison between VTM and Hevc test Model (HM) in BD-BR (a), encoding complexity (b), and decoding complexity (c).*

ration of tools are made progressively to achieve the final target of 50% BD-BR gain over the previous standard HEVC.

Figure 2.18 presents the VTM evolution from the version 2.0.1 to the version 12.1 compared with the HM through different perspectives in AI, RA, and LD-B configurations. The results are averaged through classes A1, A2, B, C, and E from the CTC. Figure 2.18(a) is the evolution of the BD-BR gain according to the VTM versions. It started on VTM2.0.1 with approximately 20% BD-BR gain for AI and LD-B configurations and 25% BD-BR gain for RA configuration. The last VTM version which is 12.1 has almost 39%, 25% and 28% BD-BR gain for RA, AI, and LD-B configurations, respectively.

Figure 2.18(b) and Figure 2.18(c) represent the evolution of complexity in both encoder and decoder side. The new tools proposed that enhanced the BD-BR also highly increased the complexity especially for the encoder. The encoding complexity is multiplied by a maximal factor of 34 compared with the HM encoder in AI configuration. The encoding complexity is less increased in RA and LD-B configurations with a maximum of 11× and 8× compared with the HM encoder, respectively. With the last version of the VTM, they managed to handle the VTM encoding complexity at 26×, 8×, and 7× in comparison to HM for AI, RA, and LD-B configurations, respectively.

The VTM software is not designed with the aim to be a real-time encoder due to the lack of optimization in terms of parameters selection, multi-threading to exploit high-level parallelism or low-level optimizations with Single Instruction Multiple Data (Simd) instruction to exploit data-level parallelism. However, reducing the complexity of the VTM while keeping the highest BD-BR gain is interesting in order to give complexity reduction directions for VVC encoder such as real-time encoder. Finally, in terms of decoding, the complexity is almost the same as the HM decoder with less than 2× in the three configurations.

This thesis has been conducted during the VVC standardization which requires to follow the VTM versions. Chapter 4 computed the complexity reduction opportunities with the VTM3.0 and the VTM10.2 under AI and RA configurations. Our first solution to reduce the complexity under AI configuration is detailed in the Chapter 6 under the VTM6.1. The second solution which exploited Convolutional Neural Network (CNN) features to predict directly split probabilities is described in the Chapter 7 and implemented in the VTM10.2. Finally, the Chapter 8 describes our complexity reduction solution achieved in the VTM10.2 under the RA configuration.

CHAPITRE 3

Related works

As previously presented, the Versatile Video Coding (VVC) Test Model (VTM) encoder brought an additional complexity due to the newly adopted tools compared with the HEVC test Model (HM). Furthermore, VVC is designed for the upcoming challenges such as ultra-high-definition, high dynamic range, or 360° omnidirectional. The high complexity of the VTM encoder plus the different formats that bring even more data than previous video formats push the complexity reduction to be a key point through the development of the new VVC standard.

This chapter describes the state of the art related to the video coding field. It details first the complexity reduction techniques related to the tree partitioning, the mode prediction, and the Multiple Transform Selection (MTS) proposed through High Efficiency Video Coding (HEVC) in Section 3.2, Future Video Coding (FVC) in Section 3.3, and VVC in Section 3.4. Complexity reduction techniques on other coding tools than the previously cited are also presented in Section 3.5. Section 3.6 gives directions and presents real-time VVC encoder and decoder. Finally, Machine Learning (ML) techniques exploited in video compression are also described in Section 3.7.

3.1 Introduction

An overview of the state of the art for complexity reduction techniques is described under three categories. Figure 3.1 illustrates the state of the art complexity reduction techniques under these three categories including tree partitioning, prediction mode, and transform selection. The categories are ordered based on their number of published papers in the literature. Several techniques are used such as complexity metric to define the complexity of a block that will be exploited to skip inappropriate depth, direction metric to limit the direction of an unlikely prediction mode, or previous information such as intra prediction modes to skip the further transform. Decision Tree (DT) and Convolutional Neural Network (CNN) are now also well established in the state of the art to reduce the complexity of encoding. Those techniques will be explained in the following sections through HM, Joint Exploration Model (JEM), and VTM software encoders.

In this thesis ML, DT and CNN terms are used to explain different approaches. Indeed, ML determines the learning algorithms that can improve automatically through experience and by the use of data. The ML term includes DT and CNN which are decision support tools exploiting tree or convolution operations.

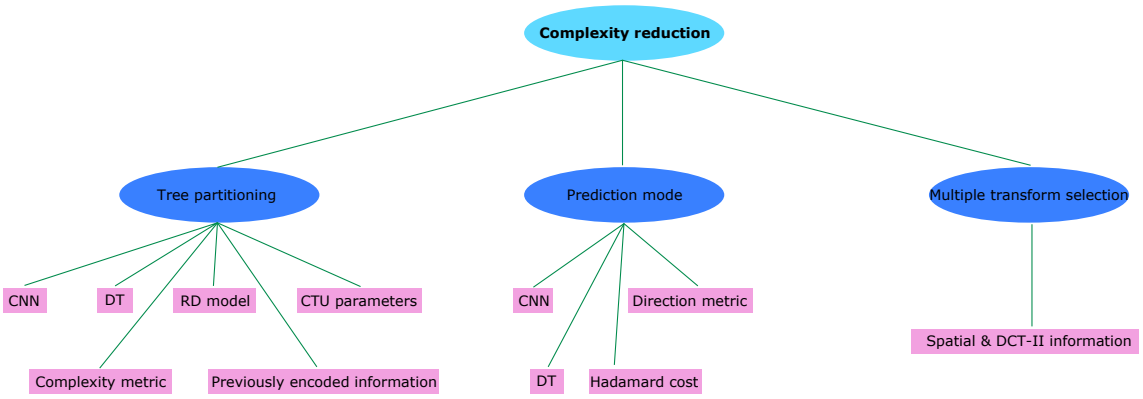


FIGURE 3.1 – *Global scheme of the state of the art for complexity reduction focusing on tree partitioning, prediction mode, and MTS.*

3.2 State of the art complexity reduction techniques for High Efficiency Video Coding

Even with less tools than VVC, HEVC complexity reduction was already a hot topic with its exhaustive search. The partitioning possibilities in HEVC are limited with only the Quad-Tree (QT) split. However, reducing the possibilities of the QT split through the depths brought an important complexity reduction opportunity. Indeed, Mercat *et al.* [28] presented an analysis on complexity reduction opportunities for the HEVC real-time intra encoder, *Kvazaar*. The impact on the energy consumption of the original video depends on the encoding parameters For the resolution and frame-rate, the HEVC encoding time is linearly depending on the number of pixels. At Quantization Parameter (QP) level, when QP is increasing, the encoding time decreases proportionally due to the Rate-Distortion Optimization (RDO) that stops earlier. They also determined the maximum complexity reduction opportunities for the tree partitioning and intra prediction by applying only the optimal solution at the encoding time. These results shown that the tree partitioning process has a potential of complexity reduction up to 78.1% whereas the intra mode prediction offers at best 30% complexity reduction.

To reduce the complexity of the HM software, most of the researchers focused on the tree partitioning due to its high complexity opportunity compared with the intra mode. Handcrafted, decision tree, and neural network techniques are detailed thereafter.

TABLE 3.1 summarizes the features and performance of the aforementioned HEVC complexity reduction techniques. Two parts of the table are depicted with the tree partitioning and the mode prediction. Sub-parts are also defined to separate AI and RA configurations. For the tree partitioning methods under AI configuration, 3 out of 4 techniques were using neural network and obtained more than 60% complexity reduction. Otherwise, for RA configuration, 3 methods were exploiting DT. Almost all methods which focused on the tree partitioning search reduced at least half of the complexity except [35] that have 34.4% but for negligible Bjøntegaard Delta Bit Rate (BD-BR) loss. For the mode prediction, most of the complexity reduction techniques were focusing on the directional intra mode.

Table 3.1 – *Main features and performance of state of the art complexity reduction techniques for HM software under All Intra (AI) and Random Access (RA) configurations.*

	Config.	Solution	Handcrafted	Decision Tree	Neural network	Software	CR (%)	BD-BR (%)
Tree partitioning	AI	Min *et al.* [29]	✓	✖	✖	HM 10.0	52.3%	0.82%
		Huang *et al.* [30]	✓	✖	✓	HM 16.5	66.7%	1.71%
		Feng *et al.* [31]	✖	✖	✓	HM 16.20	65.55%	2.02%
		Xu *et al.* [32]	✖	✖	✓	HM 16.5	62%	2.25%
	RA	Huang *et al.* [33]	✓	✖	✖	HM 16.7	70%	2.36%
		Correa *et al.* [34]	✖	✓	✖	HM 12	65%	1.36%
		Correa *et al.* [35]	✖	✓	✖	HM 16	34.4%	0.2%
		Grellert *et al.* [36]	✓	✓	✖	HM 16.8	52.4%	1.11%
Prediction	AI	Zhang *et al.* [37]	✓	✖	✖	HM 10	45%	0.8%
		Ryu *et al.* [38]	✖	✓	✖	HM 16.6	18.3%	0.5%
		Song *et al.* [39]	✓	✖	✓	HM 15.0	27.9%	1.15%
	RA	Kim *et al.* [40]	✓	✖	✖	HM 4.0	34.5%	0.4%

3.2.1 Complexity reduction of the tree partitioning

3.2.1.1 Handcrafted methods

Min *et al.* [29] proposed an algorithm to decide the Coding Unit (CU) size in AI configuration. They defined a complexity score metric that predicts the spatial complexity of a block. The global edge complexity score is computed as the l1 norm of the differences between the luminance pixel value at a position and the mean luminance value. This complexity score is computed for the two sub-blocks obtained by dividing the block in horizontal, vertical or in the two diagonals. The difference between the two complexity values of the resulting sub-blocks is computed. Another value is computed to define a local edge complexity by applying the same difference as for the global edge complexity except that a filtering is applied on the luminance pixels values and the mean luminance value. These values are then compared with predefined thresholds to decide whether the block should be split, no split, or undetermined. This solution reaches 52.3% of complexity reduction at the expense of a slight BD-BR increase of 0.82%.

Huang *et al.* [33] proposed a solution combining CU depth pre-selection, early CU and Prediction Unit (PU) terminations, and fast Transform Unit (TU) decision tree based on a Rate-Distortion (RD)-complexity optimization formula. First, a CU depth preselection is designed which calculates the RD-complexity of the fast and classical algorithm based on the complexity of both algorithms and the RD loss of the fast algorithm compared with the classical one. Several temporal and spatio-temporal features are selected to estimate the RD-loss. The comparison between the RD-complexity cost and the predefined threshold is computed in order to decide whether to skip depth 0 or not. In addition to the depth preselection, a CU depth early termination, a fast TU decision tree, and an early PU termination are proposed based on the same RD-complexity principle with other features. The features for the CU depth early termination are the number of bits generated by the motion vector difference and the mean of the absolute levels of quantized transform coefficients. The features for the fast TU decision tree and the prediction mode early termination are the mean of the absolute Hadamard transform difference and the maximum of this difference of all sub-blocks in the CU. The complexity gain of the four combined proposals under the HM16.7 in RA configuration is 46% to 70% complexity reduction for 0.48% to 2.36% BD-BR loss.

3.2.1.2 Decision Tree methods

Correa *et al.* [34] used a set of techniques to decide whether the RDO performed on the HM should be early terminated or not. For the CU early termination, three binary decision trees were available in order to predict the splitting of three CU sizes 64×64, 32×32, and 16×16. The information gain of several features such as the RD-cost for different splitting mode or the merge flag is calculated. The most valuable features are the partition which represents the PU splitting mode chosen, the neighbouring depth which is the depth of the neighbouring Coding Tree Unit (CTU), and the ratio between two RD-costs of the inter splitting modes. The second early termination is on the PU structure and starts after the merge-skip and $2N \times 2N$ modes. Four binary trees for the four CU sizes (64×64 to 8×8) are then used to predict if the other splitting modes are likely to be tested. Finally, to stop the process that determines the best residual QT structure, i.e., that splits a TU into four sub-TUs, two decision trees are trained for 32×32 and 16×16 CUs. By predicting these tree partitioning specific to HEVC, this solution achieves 65% of computational complexity reduction for 1.36% BD-BR loss.

An extension of the previous paper is proposed with online training scheme [35]. This extension focused only on the early termination of the CU based on the three decision trees for the three CU sizes. Their analysis shown that when a model is specifically trained for the sequence with its first frame, the model obtains higher decision accuracy compared to an offline training. Based on this analysis, the first frame of the sequence are encoded with the classical RDO in order to train the decision trees on-the-fly. The features are stored and a pre-processing is performed on the data to reduce the unbalancing of the split decision. This method is highly related to the temporal relationship between frames which leads to another online training for each scene change. This method, applied on the HM16 encoder in RA configuration, reaches 34.4% complexity reduction for a BD-BR loss of 0.2%.

Grellert *et al.* [36] trained a binary Support Vector Machine (SVM) to skip sub-partition evaluations. First, a feature analysis is proposed with a list of 28 features including coding flags, coding metrics such as the RD-cost, Motion Vector (MV) direction, and RD-cost calculations described in [34]. A F-score metric is used to measure how each feature contribute to the CU partitioning decision. Depending on the CU size, F-scores are depicted for the top 8 features. The prediction mode of the current CU is the best feature for the three sizes. However, several features have high F-score differences between CU sizes, for instance the number of encoded bits is the second best feature for 16×16 CUs but is not even among the eight best features for 64×64 CUs. To prevent these differences, distinct SVMs are trained for 64×64, 32×32, and 16×16 CU sizes. However, sequences with different resolutions have homogeneous F-score results among the features. This leads to the same SVM for all the sequence resolutions as it did not brought sufficient performance improvements and will reduce the applicability of the proposed method. The training of the three SVMs models considered the prediction accuracy but also the impact of the prediction on rate-distortion by optimizing the parameters with the best trade-off between RD and complexity. A threshold is applied on the output of the SVMs to define if the split is skipped or not. The value of the threshold can be modified to focus on the quality or on the complexity reduction. This method reaches 0.13% to 1.11% BD-BR loss for a complexity reduction of 34.9% to 52.4% with the HM 16.8 under RA configuration for a threshold set at 0.1 and 0.7, respectively.

3.2.1.3 Convolutional Neural Network methods

Huang *et al.* [30] developed a CNN to predict the CU depth decision. The CNN is modelled with two levels, one with the current block plus the border provided as an input and another one with only the current block which leads to different feature extractions. The partition map predicted by the CNN merges all the classification problem for the tree partitioning into one. Indeed, it directly outputs the split prediction at depth 0, 1, and 2. An heuristic method is then proposed to skip the PU partition for 8×8 CU. The heuristic method is a standard derivation-based complexity which mean the difference between the pixels of the current CU and its average pixel value. Upper and lower threshold are defined for the CNN and the heuristic method, they are symmetric with regards to 0.5. This defines the CU as no split, undetermined, or split. With those techniques, they achieve under the HM16.5 in AI configuration 1.71% BD-BR loss for 66.7% complexity reduction.

Another depth map to predict the CTU partitioning is proposed by Feng *et al.* [31]. Instead of having an output for the several depths, a matrix with numbers indicating the depth is predicted. One value for each 8×8 block inside the 64×64 CTU is outputted which reflects the local texture complexity. The CNN outputs the 8×8 matrix through convolution operation which is trained with a multi-scale L_1 loss. A conversion between the depth map and the split flags is carried-out in order to predict the depth of each block within the CTU. This technique results in 65.6% complexity reduction for 2.02% BD-BR loss under HM16.20 in AI configuration.

Xu *et al.* [32] reduced the complexity of both AI and Low Delay (LD)P configurations by predicting a hierarchical CU partition map which provides a representation of the CU partitioning. The partition map is predicted once for the partition of the whole CTU and represents the three available depths. Binary decisions are processed at each depth hierarchically to define the split of each CU. In AI configuration, the method is supported by a CNN composed of three separate normalizations that decomposed the three levels of depth for the partition map. In LDP configuration, three Long-Short Term Memory (LSTM) cells are integrated to predict the partition. These three LSTM cells are depth specific and learn the correlations of CU depths across frames. For both intra and inter configurations, a database is established by encoding and storing the partition from image and raw video sequence datasets, respectively. In AI configuration, this solution reduces the complexity by 62% for 2.25% BD-BR increase. Under the low delay P coding configuration, it reaches 54.2% of computational complexity reduction for 1.5% BD-BR increase.

Li *et al.* [41] accelerated their previous tree partition prediction method based on a CNN presented in [32]. They simplified the CNN by pruning the weight parameters at different levels to perform multiple approximations which gives flexibility for their proposed complexity-control algorithm. As the number of weight parameters varies greatly depending on the layer, the pruning influences differently each layer, i.e., a layer with fewer weight parameters is more sensitive to pruning. To manage this issue, an adaptive weight parameters pruning is designed with a parameter α that adjusts the pruning proportion at each layer. Several CNNs are trained based on the desired predefined weight parameters pruning. The un-pruned CNN requires 1.053ms per CTU for 1.549% BD-BR loss. Instead with a 0.5% weight parameters pruning CNN, the computation is shortened to 43.3µs per CTU, i.e., the CTU partition can be predicted at 45 Frames per Second (FPS) for 1920×1080 sequences, for 2.559% BD-BR loss. The complexity-control proposed in this paper works at CTU and frame levels. The CTU complexity control is based on the 7 CNNs with different weight parameters pruning and the frame complexity control focused on the error accumulation of the estimated time across the frames. Finally, the CNN run time is speed-up by 17 to

20 times with a complexity control that managed to approach the complexity desired with a difference of less than 2%.

3.2.2 Complexity reduction of prediction modes

Complexity reduction is also achievable through intra mode prediction, by reducing the set of mode to be tested.

Zhang *et al.* [37] proposed a progressive rough mode search based on the Hadamard cost to selectively evaluate potential prediction modes. A first evaluation on equally spaced four-distance nine modes plus planar and DC modes is performed to select the six best Hadamard cost modes. The modes at two-distance of the previous best mode are tested plus the selected modes of the top and left CUs if they have not already been performed. Finally, the modes at one-distance of the two best previous modes are checked along with the Most Probable Mode (MPM) which results with the best Hadamard cost mode. In addition, an early Rate-Distortion Optimization Quantization (RDOQ) skip is proposed. The first and second lowest Hadamard cost modes always go through the RDOQ. However, if the mode is at a distance of less than 2 compared with the two previous best modes, this mode is excluded from the RDOQ process. The combination of these two methods implemented in HM10 offers a complexity gain of 45% for 0.8% BD-BR loss.

Ryu *et al.* [38] proposed a random forest model to avoid unlikely intra prediction modes. Random forest is an ensemble model of randomized decision tree. In this approach, random forest model is used to classify the directional intra mode prediction. The labels are separated in 9 groups with their representative modes which represent the 35 directional modes available in HEVC. Directional block-based features are proposed to reflect the directional characteristics of a block. Four luminance values are randomly selected, one for each quadrant of the block. Then, the difference between two random values among the four are performed for the two diagonals. Several random forests are trained depending on the size of the PU, ranging from 4×4 to 32×32 with an accuracy of 66.4% to 81.0% for direction prediction, respectively. In the HM, the Rough Mode Decision (RMD) is performed to limit the prediction mode candidates to 3 and 1 compared with 8 and 3 for the original reference software. If all the RMD candidates are angular modes, the last mode is replaced by the representative mode of the predicted group in order to reduce the loss induced by the RMD. In the other case, the DC and Planar modes are added to the candidate list. This solution brings 18.3% complexity reduction for 0.5% BD-BR loss.

Song *et al.* [39] developed a deep learning-based intra prediction mode decision to skip the RMD process and to define the candidate list. A CNN is employed to define the intra mode of 8×8 and 4×4 PUs. The CNN input is the PU with its top row and left column neighbouring pixels which fit into a 5×5 block. To manage the 5×5 block for 8×8 PU, a sub-sample 2×2 filter is applied to the PU. They next carried out a coarse edge strength analysis to determine the type of the PU, i.e., flat PU, weak edge PU, and strong edge PU, through comparison with QP-based thresholds. If the PU is determined flat, DC and Planar modes are selected in the candidate list. Otherwise, if the PU is determined weak or strong, a specialized CNN is used to predict the probability of each mode and the proposed solution selects the 8 modes with the highest probabilities. Finally, a corner value is defined based on the sobel operator which will leads to a limitation of the 5 best modes if the current PU is setted to have no corner. This solution reaches 27.92% complexity reduction for 1.15% BD-BR loss.

Kim *et al.* [40] proposed a fast decision method to reduce the complexity of the intra and inter mode decision. Inter $2N \times 2N$ mode is analysed first before the skip mode for this

Table 3.2 – *Main features and performance of state of the art complexity reduction techniques under JEM software.*

	Config.	Solution	Handcrafted	Decision Tree	Neural network	Software	CR (%)	BD-BR (%)
Tree partitioning	AI	Lin *et al.* [42]	✓	✖	✖	JEM 5.0	22.8%	0.55%
		Chen *et al.* [43]	✓	✖	✖	JEM 5.0	25.4%	0.31%
		Peng *et al.* [44]	✖	✓	✖	JEM 7.0	61.4%	3.60%
		Amna *et al.* [45]	✖	✖	✓	JEM 7.0	35%	1.7%
		Jin *et al.* [46]	✖	✖	✓	JEM 3.1	42.8%	0.65%
	RA	Wang *et al.* [47]	✓	✖	✖	JEM 7.0	50.6%	1.23%
		Amestoy *et al.* [48]	✖	✓	✖	JEM 7.0	30%	0.57%
		Wang *et al.* [49]	✓	✓	✖	HM 13.0 QTBT	63.8%	1.24%
		Wang *et al.* [50]	✓	✖	✓	JEM 7.0	35%	0.55%

solution. Indeed, they used the results of inter $2N \times 2N$ mode to skip the other modes. If the best Motion Vector Differences (MVD) is equal to (0,0) and the best Coded Block Flag (CBF) is equal to 0, the early skip conditions are fulfilled, the best mode of the current PU is set as skip mode and the other modes are skipped. This solution brings a complexity reduction of 34.55% and 36.48% for BD-BR loss of 0.4% and 0.5% for in RA and LDP configuration, respectively.

3.3 State of the art complexity reduction techniques for Future Video Coding

The JEM software was developed in early 2014 to study the potential coding gain behind developing a new standard beyond HEVC called at the beginning FVC. The JEM software is based on the HM software with new coding tools such as MTS or Binary-Tree (BT) proposed to enhance the coding efficiency at the cost of higher computational complexity. The newly introduced BT tool modified the tree partitioning principle by adding the opportunity of the CUs to be rectangular. Indeed, CUs in HM were mandatory square CUs as the only available split was QT. This new opportunity brought several constraints for complexity reduction techniques. Indeed, instead of a binary classification for the QT split, the new tree partitioning design with QT and BT provided four splits with only one selected split. This new tree partitioning design has been studied and the complexity reduction methods related are further presented.

TABLE 3.2 summarizes the features and performance of complexity reduction techniques for the the aforementioned JEM. Our state of the art description focuses only on the tree partitioning with the newly introduced BT split. The AI and RA configurations are separated in the table. Under AI configuration, handcrafted methods have low complexity reduction with less than 25% for low BD-BR loss. The decision tree method obtained the highest complexity reduction with 61.4% but for a high BD-BR loss of 3.60%. Under RA configuration, Wang *et al.* [49] have the best results with 63.8% complexity reduction for 1.24% BD-BR loss but with the HM 13.0 QTBT software. All these methods brought some directions to reduce the complexity under the new tree partitioning structure for VVC.

3.3.1 Complexity reduction of the tree partitioning

3.3.1.1 Handcrafted methods

Lin *et al.* [42] focused on the BT partition to limit its complexity via spatial features. The solution is separated in two parts, the first one determines the BT depth whereas the

second one defines which BT horizontal or vertical is the most probable. For the BT depth decision, the gradient variance based on the sobel operator is computed at BT depth 1 to early terminate the split at BT depth 2 or early split into BT depth 3. Two thresholds are determined and compared with the gradient variance to early terminate the split at BT depth 2, early split into BT depth 3 if the gradient variance is lower than the first threshold, or higher than the second threshold, respectively. To limit the prediction errors, between those two thresholds, the decision is set as undetermined and the classical RDO is performed. Secondly, to decide the direction of the BT split, the difference between the gradient variance of the sub-CUs created by BT horizontal and vertical are computed which leads to a difference of gradient variance horizontal and vertical. Then, the difference of the edge between the central boundary of the sub-CUs is calculated horizontally and vertically. If the horizontal gradient variance is higher than the vertical gradient variance and the horizontal edge is higher than the vertical edge, then the BT horizontal is conducted. For the BT vertical, the same process is carried out inversely. This two contributions result in 22.8% complexity reduction for 0.55% BD-BR loss in AI configuration.

Chen *et al.* [43] proposed to early terminate the QT or BT splits depending on the neighbouring blocks. In order to early terminate the QT split, an average on the neighbouring block depths (the top-left, the top, and the left blocks) is performed. If the current QT test depth is equal to 2 and the average depth of neighbouring blocks is less than 2, the QT split is early terminated. If the current QT test depth is equal to 3 and the average depth of neighbouring blocks is less than 3, the QT split is early terminated. The BT splits also used the three neighbouring blocks as reference, the BT split is early terminated when the BT test depth is 0 and the average BT depth of the neighbouring blocks is less than 0.1. These solutions combined result in 25.4% complexity reduction for BD-BR loss of 0.31% in AI configuration.

Wang *et al.* [47] proposed a novel RD-cost estimation scheme relying on motion divergence field. Based on the estimated RD-cost, a probabilistic framework is developed to skip unnecessary splits. The distortion is defined as $D = \alpha \cdot E^{\beta}$ where α is a model parameter that fluctuates through the QP, β is a model parameter which remains almost invariant, and E is the energy of prediction errors which combines the motion and texture activities. While the encoding rate is defined as $R = R_r + R_h$ which includes R_r the entropy coding bits of residuals and R_h the header bits of side information which is a constant. The bit rate is calculated through $R_r = \gamma \cdot E$ where γ is a model parameter modified by the QP. All the model parameters are set using an offline training. They observed that the model parameters values differ slightly across the sequences meaning that they are independent of the video content. This proposed RD model estimates the RD-cost for each partition mode in order to early terminate unlikely splits through confidence interval. Indeed, a threshold is determined given a confidence interval and the probability density function of the difference between the RD-cost of the optimal partition and the minimal estimated RD-cost. The probability density function is approximated by a Gaussian distribution with the parameter σ. Two categories are created A and B which represent the split possibilities. Category A has QT and BT splits available while category B has QT split forbidden and only BT split is enabled. σ is set separately for both categories and is also content independent. The proposed algorithm reduces the complexity by 50.6% for 1.23% BD-BR increase in RA configuration.

3.3.1.2 Decision Tree methods

Amestoy *et al.* [48] proposed a solution that determines which split is the most likely to be selected between QT and BT. Random forest classifier is selected in this method to reduce

the complexity of the QTBT scheme. Indeed, a binary random forest classifier is trained offline to decide the most probable split. Spatial information and temporal information are calculated on training sequences to select and create a relevant dataset with diversity. Separate datasets are defined for the different square CU sizes from 128×128 to 16×16. Three categories of feature are analysed with features computed on the whole CU, features based on sub-quarters of the CU, and features based on inconsistency among CU sub-quarters. The first category is composed of features such as QP, pixel variance, or horizontal or vertical gradient. The second category is made of pixel variance on the CU sub-quarters or MV variance on the CU sub-quarters, for instance. Finally, the third category is based on inconsistency which is defined as $|f_1 - f_2| + |f_3 - f_4|$ for horizontal and $|f_1 - f_3| + |f_2 - f_4|$ for vertical with f_1 the top-left, f_2 the top-right, f_3 the bottom-left, and f_4 the bottom-right sub-quarter feature. The features of the third category are composed for instance of mean, variance, or gradient of horizontal and vertical inconsistency. Features are evaluated and selected through their mutual information scores. The training of these binary random forest classifiers are trained through a loss which includes the RD-cost of the optimal split and the predicted split. To reduce the RD loss introduced by misclassification, an uncertainty zone is determined where both QT and BT are tested. This solution results in 30% complexity reduction for a BD-BR loss of 0.57% in RA configuration.

Peng *et al.* [44] reduced the RDO with a three stages classification. Indeed, three binary SVMs are trained to predict if the QT, BT horizontal, or BT vertical has to be selected. Common features are extracted for the three SVMs with the texture and direction complexity, the QP, the coding bits, and the optimal RD-cost for the current CU. Independent features are also used with texture divergence between horizontal, vertical, and quad-tree partition sub-CUs for BTH, BTV, and QT, respectively. Their proposed algorithm first checks all the intra prediction modes without any split, then features are stored for the three SVMs. Following the features storage, the SVM that predicts the horizontal BT is used. If the SVM outputs the early termination of this split, the RD-cost is not calculated for BTH. The process is the same with BTV and QT, at each stage the calculated RD-cost is stored. Finally the comparison between the RD-cost is performed and the split with the lowest RD-cost is selected as the optimal one. This solution brings 61.4% complexity reduction for 3.60% BD-BR loss under AI configuration.

Wang *et al.* [49] dynamically derived the partition parameters at CTU level and designed a joint-classifier decision tree structure at CU level. The first part of the proposed solution adapts the partition parameters, i.e., the minimum QT size, the maximum BT size, and the maximum BT depth in order to adapt to the local characteristics of the CTU. Minimum QT size and maximum BT size are derived from the average size of QT leaf nodes in the neighbouring CTUs. Maximum BT depth is set according to the ratio between the average size of QT leaf nodes and BT leaf nodes in the neighbouring CTUs. The second part focuses on the CU level with a joint-classifier method. Two classifiers are used, the first classifier decides whether to conduct the QT or not and the second classifier decides whether to conduct the BT or not. The classifier inputs which are the block size, the Lagrange multiplier, a texture value, the ratio between horizontal and vertical texture values, the average depth for QT and BT among neighbouring blocks, and the neighbouring splits are selected based on their information gain performances. Four decisions are available based on the outputs of the two classifiers combined. If both classifiers predict the early termination, the RD-cost of the current block is performed and the tree partitioning is stopped. If the QT classifier outputs QT and the BT classifier predicts to stop the BT, then the RD-cost of the current block is calculated and only the QT is conducted. The same process is applied in the other way with BT predicted and not QT. Finally, when both

classifiers predict its splits, both splits are checked. This method brings 63.8% complexity reduction for 1.24% BD-BR loss in RA configuration.

3.3.1.3 Convolutional Neural Network methods

Amna *et al.* [45] utilized a CNN to reduce the QT module partition. Three CNNs are proposed to cover the three levels of CU from 128×128 to 32×32. A dataset is established for the three CU sizes with raw image database. The three CNNs are working in parallel to predict a map of the whole CTU that represents the QT partition. The input of the CNNs is the luminance pixels of the current CTU preprocessed for each CNN with a down-sampling to match the CU's smaller depths. Before the fully connected layer, the vectors from the three models are concatenated to obtain the information from all the CNNs. Then, separate fully connected layers output maps of probabilities that are used to predict the QT depth of each block inside the CTU. This solution results in a complexity reduction of 35% for 1.7% BD-BR loss under AI configuration.

Jin *et al.* [46] addressed the QTBT partition problem with a multi classification CNN. The multi classification problem focuses on the maximum QTBT depth within a 32×32 block. A distribution of the partition depth is presented which is highly unbalanced with depth 4, 7, 9, and 10 which have the highest percentage. Based on this distribution, 5 classes are created which represent the minimum and maximum QTBT depth. The CNN architecture is similar to ResNet [51] and is trained under the L2 loss combined with a misclassification penalty term. The CNN is included in the JEM and performs the 32×32 patches of the CTU. First at CTU level, if the classification of all patches is the lowest depth range, RDO calculation is performed. Otherwise, the 128×128 RD-cost calculation is skipped and the RDO goes directly to the four sub-CUs. At 64×64 level, if the prediction of all patches are under the medium class, the RD-cost of the CU is performed. Otherwise, the 64×64 RD-cost calculation is skipped and the RDO goes directly to the four sub-CUs. Finally, at 32×32 level, the JEM calculates the RD-cost for each partition until the depth is not included in the predicted depth range. This method brings 42.8% complexity reduction for 0.65% BD-BR loss under AI configuration.

Wang *et al.* [50] proposed to reduce the inter coding complexity of the JEM through CNN QTBT prediction. Instead of predicting the most likely split at each depth, they predict the maximum depth level. The depth of a CU is calculated via $Depth_{CU} = 2 \times QTDepth_{CU} + BTDepth_{CU}$ with $QTDepth_{CU}$ and $BTDepth_{CU}$ the QT and BT depth of the CU, respectively. The $Depth_{CU}$ ranges from 0 to 10 with the minimum QT size defined at 16×16 and the maximum BT depth defined at 4. 6 classes are defined from very flat with a maximum depth set as 0 to deep texture with a maximum depth set as 10. The maximum depth distribution is highly unbalanced at CTU size with half of the maximum depth is determined as 10. If the prediction was performed under CTU size, half of the CTU may not provide any time saving. The maximum depth distribution of 64×64 and 32×32 were also analysed and resulted in more homogeneous distribution which is more appropriate for training. The CNN input selected is the 64×64 residual block after the motion compensation application and the output is a multi-class classification with the 6 classes presented previously that represent the maximum depth of the input block. Finally, this solution included in the JEM is divided in four steps. The first one divides the current CTU in four 64×64 blocks and the maximum depth is predicted for each patch. The second step applies the same process but for the co-located CTU in the reference frame. The third step adjusts the maximum depth of the current patches with $depth = depth + max\{Depth'_{co} - depth'_{co}, 0\}$ where $Depth'_{co}$ is the actual co-located maximum depth and $depth'_{co}$ is the predicted co-located maximum depth. This maximal

depth adjustment reduces the risk of false prediction. The last step uses the maximum depth predicted by the CNN which is adjusted with the co-located partitioning information to control the current CTU partitioning. This method offers 35% complexity reduction for 0.55% BD-BR loss under RA configuration.

Under the development of the JEM, Galpin *et al.* [52] proposed a new split with asymmetric BT. Moreover, they developed a CNN to limits the potential split choices and maintain low complexity. The asymmetric BT proposition split the CU into two sub-CUs that can be on top, bottom, left, and right with the ratio 1 :3 or 3 :1. The number of luminance blocks tested with the QT plus BT is 22× more than only QT and the integration of the asymmetric BT is 3× more than QTBT without heuristics. This analysis brought the necessity of reducing the complexity to make the proposed split viable and they wanted to demonstrate that CNNs are capable of resolving the split selection problematic with reasonable BD-BR loss. As many splits are available, they adapted the CNN output to cover the whole 64×64 block partitioning search. Indeed, the output is a vector that represents the whole 64×64 partitioning with a probability for each 4×4 block boundary. At each block size, a calculation based on these probabilities is computed to propose a probability for each split. Based on these split probabilities, a threshold is applied to determine which split is performed or skipped. With the CNN, the number of luminance blocks tested with the QTBT and asymmetric BT is reduced to 68% compared with the one with no heuristics. At similar BD-BR performance, the complexity of this solution added with the asymmetric BT is 21% of the JEM encoding with QTBT. For the same complexity, a BD-BR gain of 6% is achieved under AI configuration.

3.4 State of the art complexity reduction techniques for Versatile Video Coding

VVC replaces the FVC name and adopted new tools in order to achieve higher coding efficiency than previous state of the art standards. As previously presented, the VTM complexity is much more higher than the HM complexity. Complexity reduction under HM was already a hot topic with all the techniques presented to limit the exhaustive search of the tree partitioning or the mode prediction. Under the VTM software the complexity is again a key aspect, so, tackling the VTM complexity is crucial for its application. Compared with the JEM software, another split is introduced with Ternary-Tree (TT) which has a specific ratio for its sub-CUs. The representation of the partition structure is one of the main issue to limit the tree partitioning. Indeed, using techniques that have small computational complexity is also critical. The next analyses present the performance of the VTM focusing on the complexity.

Bossen *et al.* [53] described the VTM specification with its several modules starting with the tree partitioning and finishing with the in-loop filters. Compared with HEVC, numerous new tools have been included in VVC and are presented in this paper [53]. In addition, an analysis on the VTM10.0 is proposed focusing on its complexity at both encoding and decoding side compared with the HM16.23. The relative run time at encoding compared with HM for A class sequences is highly increased with a multiplication by 25 and 9 under AI and RA configurations, respectively. The QP has also an high impact on the encoding time particularly with low QP values such as 22. The encoding time can be multiplied by 40 times compared with HM under AI configuration. A study on the different modules is performed for encoding. Indeed, the encoding is divided into modules that have different running times depending on the complexity of the tools. The three highest

encoding complexity modules are intra, merge, and quantization. However at decoding side, the run time increase is significantly lower with a maximum increase of a factor of 2. Some Single Instruction Multiple Data (SIMD) optimizations are included in the VTM. When deactivating them the impact on run times under AI configuration is at a maximum of 10% and 42% augmentation at encoding and decoding side, respectively. Under RA configuration, the complexity almost doubled at encoding and decoding side. Through all those investigations, they defined several tools that need to be optimized in order to reach real-time software encoders. Across the tools, the tree partitioning module is the most expensive in terms of complexity as it is performed at the top of other modules.

Authors in [7] detailed the complexity of each tools independently. Multithreading can also further reduces the encoding time, for instance, with 6 threads a speed-up of about 4× is achievable. Francois *et al.* [7] evaluated the VVC tools performance compared with HEVC. First, a comparison between VTM8.0 and HM16.19 is given with the Peak Signal-to-Noise Ratio (PSNR), Video Multimethod Assessment Fusion (VMAF), and Multi-Scale Structural Similarity (MS-SSIM) metrics. For a selected metric, the average bit-rate reduction is depicted along with the complexity at encoding and decoding. Under AI configuration, the saved BD-BR with PSNR, VMAF, and MS-SSIM are 27.3%, 27.4%, and 26.8%, respectively for 25× encoding time and 2.5× decoding time. Under RA configuration, the BD-BR saved with PSNR, VMAF, and MS-SSIM are 40.1%, 41.4%, and 37.4%, respectively for 9.5× encoding time and 2.5× decoding time. PSNR and VMAF have homogenous results when MS-SSIM has lower results than the two others metrics. Subjective observations are also proposed with experts that confirmed the outperforming quality of VTM compared with HM for the same bitrate. However, depending on the codec, different visual impact are reported such as textures that trend to be smoother for VVC. They also measured each tool by switching them off independently with Common Test Conditions (CTC) sequences and non-CTC sequences. The tools that brought the most BD-BR loss when disabled is the new BT and TT splits with more than 12%. Adaptive Loop Filter (ALF) and cross-component linear model were the second and third tools with 8% and 4% BD-BR losses, respectively. Using CTC or non-CTC sequences brought very similar results for most of the tools, the maximum difference was for ALF with 2.2% BD-BR gap. Nonetheless, the BT and TT partitioning splits brought important BD-BR savings but at a high encoding complexity, when disabled the encoding time is reduced by 6×. All the other tools were between 80% and 100% for encoding time. Under decoding, when disabling each tool independently, the complexity is reduced at a maximum of 80% of the reference decoding time.

3.4.1 Complexity reduction for tree partitioning

TABLE 3.3 summarizes the features and performance of the aforementioned VVC complexity reduction techniques. The complexity reduction techniques addressing the tree partitioning, the intra mode prediction, and the MTS are shown in this table. For the tree partitioning, methods under AI and RA configurations are displayed. Under AI configuration, the methods have similar results ranging from 33.4% to 53.2% complexity reduction with BD-BR loss proportional to the reduction. Under RA configuration, the DT method surpassed the other [63] with 38.6% complexity reduction for 0.97% BD-BR loss. Finally the methods reducing the complexity of the intra mode prediction and the MTS were all handcrafted methods fluctuating from 23% to 30.1% with less than 0.6% BD-BR loss.

Table 3.3 – *Main features and performances of state of the art complexity reduction techniques under VTM software.*

	Config.	Solution	Handcrafted	Decision Tree	Neural network	Software	CR (%)	BD-BR (%)
Tree partitioning	AI	Chen *et al.* [54]	✓	✖	✖	VTM 4.0	53.2%	1.62%
		Cui *et al.* [55]	✓	✖	✖	VTM 5.0	51%	1.34%
		Fu *et al.* [56]	✓	✖	✖	VTM 1.0	45%	1.02%
		Chen *et al.* [57]	✖	✓	✖	VTM 2.1	51%	1.54%
		Yang *et al.* [58]	✖	✓	✖	VTM 2.0	52.6%	1.56%
		Tang *et al.* [59]	✓	✖	✓	VTM 5.0	33.4%	0.99%
		Zhao *et al.* in [60]	✓	✖	✓	VTM 7.0	39.4%	0.86%
		Li *et al.* [61]	✖	✖	✓	VTM 7.0	45.8%	1.32%
	RA	Tang *et al.* [62]	✓	✖	✖	VTM 4.0.1rc1	31.4%	1.34%
		Amestoy *et al.* [63]	✖	✓	✖	VTM 5.0	38.6%	0.97%
		Pan *et al.* [64]	✖	✖	✓	VTM 6.0	24.8%	2.52%
Pred.	AI	Yang *et al.* [58]	✓	✖	✖	VTM 2.0	25.5%	0.54%
		Zhang *et al.* [65]	✓	✖	✖	VTM 4.0	30.1%	0.58%
MTS	AI	Fu *et al.* [66]	✓	✖	✖	VTM 3.0	23%	0.16%

3.4.1.1 Handcrafted methods

Tang *et al.* [62] proposed two separate fast block partitioning method for both intra and inter coding. The intra solution focused on the texture of the block to early terminate the different splits. The edge map is obtained through the block-based Canny algorithm and the vertical and horizontal edge are calculated according to this edge map. Indeed, they extract an edge map that is used to define if the tree partitioning search is stopped. If the tree partitioning search continues, a comparison is performed between the horizontal and vertical edges. The larger value defines the most probable split. For the inter solution, a metric is calculated based on the pixel value difference between the current frame and its reference frames. This metric is therefore a ratio between the number of pixel difference equal to 0 with the total number of pixel of the current CU. It is then compared with a fixed threshold in order to decide the early termination of the block. Otherwise, the previous block-based Canny algorithm is used to reduce the tree partitioning search. Under AI configuration with its specialized solution, the complexity reduction is about 36.2% for 0.71% BD-BR loss. Under RA configuration, the solution offers 31.4% complexity reduction for 1.34% BD-BR loss.

Chen *et al.* [54] proposed a fast partition decision relying on variance and gradient. Three different steps are necessary for this method which focuses on 32×32 block. The first step computes the variance of the pixels inside the block and compares it with a predefined threshold to early terminate the tree partitioning search. The second step decides whether QT should be performed and the other splits early terminated on the 32×32 block. They assess the sum of absolute gradients of each pixels under horizontal and vertical directions with the Sobel operator. Then, a ratio is quantified by dividing the bigger sum of absolute gradient in horizontal or vertical by the other. If this ratio is lower than a predefined threshold and if both sums of absolute gradient are superior than another predefined threshold, then the block is split with QT and directional splits are not processed. Finally, the third step computes the variance of the variance of the sub-blocks created by the several splits. The chosen split is selected when it has the maximum variance compared to the other splits and it is the only one performed. The thresholds are set empirically through a sequence composed of images from the DIV2K dataset [67]. This three steps solution reaches 53.2% complexity reduction for 1.62% BD-BR loss under AI configuration.

Cui *et al.* [55] developed a gradient-based early termination algorithm to skip unnecessary splits. Firstly, for CUs larger or equal to 16×16, the partition is stopped if the split flag proposed is down to 0 or the intra prediction of this block is not performed. Then, the RDO goes directly to the sub-blocks if the flag is up to 1. To determine this split flag, the gradients of the sub-blocks are computed and compared with thresholds. If the gradient ratios are superior or inferior to predefined thresholds, the flag is set to 1 or 0, respectively. Otherwise if the split flag is equal to -1, the RDO is processed normally. Secondly, the selection of the split direction is proposed on the leaf node blocks of the QT. A comparison between the gradients of four direction with horizontal, vertical, and the two diagonals is performed to determine which of the horizontal or vertical split direction will be avoided. Finally, to determine which of the BT or TT is the most probable split when the split direction is not decided, a flag for BT skip and TT skip are set to one when the gradients ratios are inferior to the same threshold as the first step. This method enables 51% complexity reduction for a BD-BR loss of 1.34% under AI configuration.

Fu *et al.* [56] proposed two distinct fast block partition techniques through Bayesian decision rule. Several analyses were conducted on the correlation between the current CU and its sub-CUs. The probabilities of vertical splits and TTH for the current CU based on the splits chosen for the sub-CUs are presented. For instance, when BTH are selected for both sub-CUs, the probability of vertical splits are 5.4%. Another feature is investigated with the optimal intra mode of sub-CUs and its correlation with the current CU split type. For instance, they observed that the probability of vertical splits are low when one of the sub-CUs selects an horizontal direction for intra mode prediction. Relying on these analyses they used both optimal intra mode and split of the sub-CUs to skip uncertain splits. First, after BTH is evaluated, a Bayesian rule decides to early terminate the vertical splits. Indeed, a comparison is processed to determine the skip of the vertical split which depends on the probability density function estimated off-line. To balance the trade-off between the complexity reduction and the RD-cost, a factor is added to the comparison based on the detection rate and the hit rate of the early skip. The second technique intends to skip TTH split with the information provided by the vertical splits and depending on both sub-CUs partitioning and intra prediction mode. Indeed, if the sub-CUs splits are BTV and the sub-CUs plus the current CU intra modes are in vertical direction for intra mode prediction then TTH is skipped. Furthermore, if the BTV RD-cost is inferior to the BTH RD-cost then TTH is also skipped. Those techniques reaches 45% of computational complexity reduction for 1.02% BD-BR increase.

3.4.1.2 Decision Tree methods

Amestoy *et al.* [63] designed a cascade framework through random forest classifiers to determine the probability of each split. Three classifiers are included inside the cascade framework. The first one focuses on the determination of split or no split. If the split is selected, the second classifier predicts if the split is QT or one of the directional splits. Finally, if the directional split category is predicted, the horizontal or vertical splits are preferred. In order to improve the accuracy of their classifiers, the impact of each feature is evaluated such as QP, variance, and mean of the block gradients for the three different classifiers. As a result, relying on the classifier, the selection of features is different based on their motion information and their improvement of the classification rate. The training of these classifiers is impacted by the RD-cost errors caused by false prediction. Furthermore, the thresholds applied to the different classifiers are optimized. A risk interval is proposed to limit the RD-cost increase by computing both splits of the classifier output when the probability falls in this risk interval. In RA configuration, this solution enables from 25.5%

to 38.6% of complexity reduction for respectively 0.43% to 0.97% BD-BR loss depending on the risk interval configuration.

Chen *et al.* [57] presented a SVM that are trained online to classify the CU splitting. Six SVM are defined to predict the CU from size 32×32 to size 16×8, or 8×16. If the frame is an online training frame, the features are extracted with the direction of the split, i.e., horizontal or vertical. Then the models are trained and extracted to make it available for the further prediction. To take advantage of the online training and to adapt the switching scene, the online training frame is set at each 80 frames. New features are proposed to correspond to the horizontal and vertical splitting. They introduced the difference between entropy variance of BTH and entropy variance of BTV, the difference of entropy variance between two directions with TT, the texture contrast difference in horizontal and vertical splitting, and a last feature relying on the horizontal and vertical coefficient of the Harr wavelet transform. Therefore, to estimate the importance of features, the F-score is calculated. Homogenous results among the sequences and the features are depicted. The classifiers accuracy is higher than 80% on average among the CU sizes and the sequences. This method reaches 51% of computational complexity reduction for 1.54% BD-BR increase.

Yang *et al.* [58] reduced the tree partitioning exploration via a cascade framework. Observations are deduced from experimentations on the QT depth and BTTT depth. 71% of CTUs are encoded with QT depth inferior to three and 21.8% of CTUs are encoded with QT depth equal to 1. The percentage of large CU size is increased when sequences are encoded with large QP. Depth distribution is highly related to the sequence resolution. Indeed, high definition sequences have higher probabilities to have large CUs. Low definition sequences have higher probabilities to have small CUs. Inside the same resolution, the depth distribution is related to the contents. Based on this analysis, a cascaded framework is designed to accelerate the tree partitioning scheme and to preserve the coding quality. Three categories of feature are included for the decision tree with global texture information, local texture information, and context information. Global texture information is composed of five features that measures the content homogeneity and direction of the processing CU. Local texture information concentrates on the differences between the sub-blocks through the local texture variance. Context information is related to the number of neighbouring CUs that have more QT depth and QTBTTT depth. Finally, the decision trees cascade framework starts with the QT prediction. If QT is selected, then other splits are early skipped for this depth. Otherwise, the other decision trees are predicting their split probabilities independently. This method brings 52.6% complexity reduction for 1.56% BD-BR loss under the AI configuration.

3.4.1.3 Convolutional Neural Network methods

CNN is taking a lot of importance for the image and video coding field. Indeed, they were highly used to limit the RDO search for HEVC and FVC.

Pan *et al.* [64] trained a multi-information fusion CNN to early terminate the tree partitioning. The problem is modelled as a binary classification with 0 as a split and 1 as a no split, i.e., an early termination. The inputs of the CNN are the luminance pixels, the residuals, and the bi-directional motion-field of the current CU. A limitation on the CU size is applied to limit the CNN inferences when the CU size is lower than 16×16. To manage the size upper than 16×16, the CNN is a fully convolutional network plus three asymmetric kernel convolution group that are used to enhance the accuracy on the three categories of size. The loss used to train the CNN is customized with a classical cross-entropy loss plus penalties on incorrect prediction with larger RD-cost loss. Finally,

they defined experimentally the thresholds with one for 128×128 CU and one for the other CU sizes. With this method, they obtain 24.8% complexity reduction for a BD-BR loss of 2.52% under RA configuration.

Tang *et al.* [59] proposed an adaptive CU split decision based on a pooling variable CNN that targets different CU shapes. With this adaptive CU size, only one CNN is necessary to compute their method. For square or rectangular CU, the gradient of the residual block is calculated and set as input in order to predict the split or not. In previous work on HEVC with CNN, in order to put all CU sizes as input, a simple downsampling was necessary as the block was always square. To downsample the block with VVC, they proposed to take advantage of the pooling layer to get the same input feature size at the fully connected layer. This pooling layer is interesting as no parameters need to be trained, so, based on the input residual block size, the pooling layer size vary. Finally, before the fully connected layer is processed, the QP, width, and height are added as a neuron. Before applying the CNN, a pre-decision algorithm is proposed to skip the CNN prediction when the results of splitting are easily manageable. This pre-decision is based on the calculation of gradient through the Sobel operator on the current residual block. If the gradient result is inferior at $QP^2 * 0.15$ then the early skip is chosen, however if the gradient result is superior at $QP^2 * 9$ then the split is computed and the sub-CUs are directly checked. This counts as 5% of all the CU tested. Both techniques combined results in 33.4% complexity reduction for 0.99% BD-BR loss under AI configuration.

Another CNN model is proposed by Zhao *et al.* in [60]. This solution also developed an heuristic method to determine if a CU is homogeneous. Homogeneous region will lead to early skip of the partition and is defined with the standard deviation which represents the texture complexity. A threshold is established empirically depending on QP and CU depth which are highly related to the texture complexity of the CU. The standard deviation is compared with the threshold. If the standard deviation is lower than the threshold then the CU is considered as homogeneous and no split is further performed. However if the standard deviation is higher than the threshold then the CNN is applied to predict the further partition. The first CNN has an adaptive structure which allows to have one CNN for the different CU sizes with the max pooling layer. As input the top-left, top, top-right and left neighbouring residual blocks are utilized. However, if the CU is an edge CU, the reference blocks are not all available which makes impossible the use of the adaptive CNN. To deal with this problematic a multi-feature fusion CNN is introduced which uses the standard deviation and the QP to determine the partition. This solution achieves 39.4% of computational complexity reduction for 0.87% BD-BR loss.

Li *et al.* [61] proposed a deep learning approach to predict the CTU partition with a binary or multi-class classification at each CU depth. This deep learning approach is relying on a multi-stage exit CNN to adapt the CU partition search. The multi-stage exit CNN predicts the whole 128×128 luminance CTU partitioning. Indeed, the multi-stage exit predicts at each depth the split or not, i.e., for instance if four 64×64 blocks are available and the CNN predicts a QT split for the top-left 64×64 block and no split for the others, it leads the CNN to continue its prediction on the sub-networks for the four 32×32 sub-blocks and exit the three others 64×64 blocks. Each block goes through the principal network or the sub-networks to obtain the split mode. This leads to a global partition determination of the whole CTU by a one-pass CNN prediction. To train the CNN, they designed an adaptive loss function based on the cross-entropy that penalizes the split proportion and also the high difference of RD-cost between the predicted split and the optimal one. This technique enables to reduce the complexity by 45.8% for 1.32% BD-BR loss.

3.4.2 Complexity reduction of intra mode prediction and Multiple Transform Selection

For the VVC standard, several works have been proposed to limit the search space for the intra mode prediction or the transform module.

Yang *et al.* [58] proposed additionally to their partition structure decision, a fast intra mode decision that reduced the number of modes using a one dimensional gradient descent search. The initial search mode is first determined with the MPM as 70% of the blocks select one of them. The MPM are organized based on their Hadamard cost and the minimal one is selected. The search pattern is modified to be bidirectional, the left and right gradient descent searches are achieved successively and two modes are returned. They are compared so the mode with the minimal Hadamard cost is retained. The neighbouring modes of this previous mode are finally checked, the Hadamard cost is then performed on $[Mode - 2, Mode + 2]$. Finally, as DC and planar modes have high probabilities to be the optimal mode, they are added to the last directional mode with the lowest Hadamard cost to be checked under the full RDO. This solution brings 25.5% complexity reduction for a BD-BR loss of 0.54% under VTM2.0 in AI configuration.

Zhang *et al.* [65] fastened the intra mode decision algorithm relying on texture region features. This work relies on the high correlation between the prediction mode and the pixel similarity in the corresponding direction. To limit the intra prediction mode evaluation, 4 classes are defined which correspond to 0°, 45°, 90°, and 135°. Planar and DC modes are added on all those 4 classes to enhance the accuracy based on experimental statistics. The Canny operator is proposed to select one of those classes by exploiting the gradient of each pixel. Those horizontal and vertical gradients are used to calculate the gradient amplitude and the gradient angle. The gradient amplitude and angle are then utilized to calculate the texture direction for the four previously defined directions. Finally, according to all the previous calculations, an energy is determined for each direction. By comparing those energies, the larger one is determined as the optimal one. However, if the second energy is higher than 0.8× the first one, both directions are checked. This technique allows 30.1% complexity reduction for 0.58% BD-BR loss under VTM4.0 in AI configuration.

Instead of reducing the tree partitioning scheme or the intra mode prediction, Fu *et al.* [66] proposed to focus on the MTS to reduce the complexity. Two processes are defined to limit the MTS computation. Firstly, a statistical analysis detailed the distribution of the ratio between the RD-cost of the primary transform and the MTS for the last children sub-CU. It is demonstrated that the ratio is distributed near 1 which allow them to skip the calculation of the MTS as it can be approximated from the Discrete Cosine Transform (DCT) with a factor α. Indeed, to skip the MTS of the last children sub-CU, the RD-cost of the parent must be inferior or equal to the RD-cost of the sub-CUs plus the RD-cost of the last children encoded only with DCT and multiplied by the factor α which is predefined from experimental results. Secondly, an early termination scheme of the MTS is proposed relying on the intra prediction modes. The DCT-II is first applied on all the modes, then a different order of importance for the MTS candidate is defined based on their frequencies of apparition on neighbouring blocks. When the RD-cost of a defined intra mode prediction and a defined MTS candidate is lower than the other RD-costs that have already been calculated, then the following MTS calculation is skipped. This method results in 23% complexity reduction for 0.16% BD-BR loss in the luminance component under the VTM3.0 in AI configuration.

3.5 Complexity reduction of coding tools

In this thesis, we focus on the VVC complexity reduction related to the tree partitioning, the prediction mode, and the MTS. However, different parts of the standard can be refined to reduce the complexity of the encoder or decoder.

3.5.1 Transform

Hamidouche *et al.* [68] proposed a hardware-friendly Discrete Sine Transform (DST)7 and DCT8 approximations applied to VVC. The MTS highly increases the complexity of the transform module with the selection of the adequate transform. DCT8 can be calculated from the DST7 involving only a vector reflection matrix and a sign changes matrix. They focused on the approximation of the forward DST7 and inverse DST7 which are based on DCT2 fast implementations. To compute an approximation of the DST7, a genetic search algorithm is computed to find the adjustment band-matrix that is necessary to compute the DST7 approximation based on the DCT2. This algorithm changes individual elements of the adjustment matrix in the mutation process until it converges. They reach a limited coding loss with 0.07% BD-BR for the AI configuration with 96% encoding time and 85% decoding time. For RA and LDB, the coding loss and the complexity in encoding and decoding remain the same. However, the gain in number of operations and in memory usage is significant for hardware implementation such as field-programmable gate array (FPGA) platform.

3.5.2 Quantization

Wang *et al.* [69] reduced the complexity of the trellis-coded quantization method. It is implemented as dependent scalar quantization to maintains simultaneously two quantizers with four transition states. Based on the RD model, the trellis departure point is first determined to lower the number of trellis stage. If the absolute value of the transform coefficients is lower than a threshold based on the rate and the quantization step size, the associated quantization coefficients are determined as 0 which postponed the trellis starting point to the next non-zero coefficient. With the trellis-coded quantization, an high encoding complexity cost is added due to the RD-cost calculations with the trellis branches to optimize the quantization. To reduce this added complexity, unlikely quantization candidates are removed to suppress the associated transition routes. Three different cases are determined by the absolute value of the scalar quantization at a specific position. If this value is equal to 0, the quantization candidates with higher quantization levels are removed without RD-cost calculation. If the value is equal to 1 or 2, two possibilities appear the savings of coding bits or the increase of coding bits. The second scenario removes larger quantization candidates from the trellis graph. Finally, if the value is over 2, the candidate level 0 is not tested. These techniques achieve 11% and 5% encoding time savings with 0.11% and 0.05% BD-BR increase under AI and RA configurations, respectively.

3.5.3 Parallelization

Amestoy *et al.* [70] processed the VVC encoding in parallel. This method benefits both from the dynamic tile and the rectangular slice partitioning. The partitioning is separated in two parts, the first step is the encoding time minimization. It calculates a minimum estimated time based on the CTU times of the co-located frame. This estimated time is the time constraint to the second step which computes the rectangular slice clustering. Indeed, the

estimated time of the frame is bounded with a Lagrangian parameter that either minimizes the estimated time or optimizes the encoding quality through the partitioning. When only minimizing the encoding time with 8 threads on Ultra High Definition (UHD) sequences, this method reaches a speed-up of 5.43× VVC for 2.49% BD-BR loss. When offering a trade-off between encoding time and encoding quality, this method obtains a speed-up of 5.34× VVC for 2.33% BD-BR loss.

3.6 Real-time Versatile Video Coding encoders and decoders

VVC is a standard which needs encoders and decoders to be popularized. Indeed, a major part of its development comes through real-time encoders and decoders with optimal performance. All these previous techniques which intended to reduce the complexity of the standard give directions for real-time encoding and decoding.

3.6.1 Study and directions for real-time Versatile Video Coding encoding

A first reflection is conducted on the real-time VVC encoding with Brandenburg *et al.* [71] throughout an optimized encoder. A study on the tools specified in the configuration file is proposed which includes for instance the tree partitioning or the ALF. The depth of QT, BT, and TT are important parameters to reduce the complexity. They proposed different configurations of depth, the deepest depth without the fast QTBT tool achieves 15.74% BD-BR savings for 173% complexity against the HM. Another trade-off halves the complexity for 11.22% BD-BR savings with the content based fast QTBT search strategy activated. The other tools are analysed independently with their impacts on the coding performance and encoding time. The BD-BR savings range from 11.22% to 35.38% for a complexity of 50% to 291% against HM, respectively. Thereafter, an optimized VVC software encoding is proposed by supporting a subset of tools included in the VTM. Additional Simd implementations are added for transform, in-loop filtering, and small block interpolation filtering which altogether at bit-equal runs 33% faster than the VTM. A modification on the VTM tree partitioning early termination is proposed to take into account previous splits at the same depth rather than sub-splits due to the maximal depth of 1 for BT and TT in non-intra frames. The affine merge mode is skipped for non-reference frames and affine motion estimation is tested only if the affine merge mode has a lower RD-cost than the merge mode. A reduction on the exhaustive tests of adaptive motion vector resolution, merge with motion vector differences, symmetric motion vector difference, and motion estimation is also proposed. All these optimizations enable the proposed optimized software to speed-up the VTM by 22.8× for a BD-BR loss of 12.71% under RA configuration.

3.6.2 Practical Versatile Video Coding encoder

Wieckowski *et al.* [72] implemented an open and practical VVC encoder. The software is based on the VTM with 5 presets that represent different coding quality and complexity operating points. Parallel processing performed in this work obtains a great scaling with 10 threads for HD sequences and 20 threads for UHD sequences. Moreover, an adaptive quantization mode is proposed to enhance the perceived quality based on the extended Psnr measure [73] to distribute the bitrate among and within the frames. With UHD sequences, the last version of the optimized encoder (v1.0.0) obtains a speed-up of 142× and 81× the HM for a BD-BR savings of 13% and 28.2% with the faster and fast configurations, respectively.

3.6.3 Real-time Versatile Video Coding decoder

Zhu *et al.* [74] developed a real-time VVC software decoder. Such as the previous optimized VVC encoder, the real-time VVC decoder includes SIMD optimizations. The tools considered for SIMD optimizations under 16 bit and 8 bit are intra prediction, inter prediction, inverse transform, inverse quantization, ALF, cross-component ALF, deblocking filter, Sample Adaptive Offset (SAO), and Luma Mapping with Chroma Scaling (LMCS). These optimizations speed-up the several tools independently from 1.22× to 18.1×. Furthermore, multiple threads are available to run the decoding process in parallel at several levels including frame, CTU, and tasks such as in-loop filtering and sub-CTU level. For UHD sequences with single thread, the decoding process goes from 5.7 FPS to 10.1 FPS at QP 22 and 37, respectively. However, with 8 threads, it decodes UHD sequences from 38.6 FPS to 64.7 FPS at QP 22 and 37, respectively.

3.7 Machine Learning in video coding

Through the last decades, ML has brought a breakthrough in various fields. Indeed, the ML has contributed to many technological advances, especially in the image and video domains with the CNN development. Obviously, ML has also lead the video coding field to exploit this technology through different aspects.

3.7.1 End-to-end video coding frameworks

Although the standards such as HEVC, VVC, or AOMedia Video 1 (AV1) are the state of the art in video coding, other frameworks are started to emerge with the new alternatives brought by machine learning and especially neural network. These end-to-end video coding frameworks are different than conventional video coding standards which are only based on handcrafted modules. Indeed, the key components in video coding such as motion compensation, quantization, or bitrate estimation are implemented with end-to-end neural networks on these new frameworks.

Lu *et al.* [75] proposed an end-to-end deep video coding framework with a one to one correspondence between the conventional video coding modules and their respective neural networks. Firstly, a CNN model predicts the optical flow which is the motion information. Secondly, based on the previous optical flow, a motion compensation network is designed to obtain the predicted frame. Thirdly, the transform module is replaced by an auto encoder composed of an encoder and a decoder for transform and inverse transform, respectively. These networks better exploit the non-linearity compared with the classical transform. The quantization module is replaced by a quantization process proposed in a end-to-end image coding technique [76]. Finally, the bitrate is estimated with a CNN inspired by [77]. This end-to-end deep learning framework outperforms Advanced Video Coding (AVC) in terms of coding efficiency, however, it reaches 24 FPS for encoding a 352×288 video which is very low compared with the AVC commercial software which is able to perform 250 FPS.

Rippel *et al.* [78] proposed an end-to-end video coding framework for low latency mode with a generalized motion estimation and an ML-based spatial rate-control. Indeed, they simulate the traditional flow-residual pipeline with ML techniques. They encode jointly the flow and residual with an encoder that takes as input the last reconstructed frame, the current target frame, and a learned state which is an accumulation of temporal information through recursive updates. Moreover, a ML-based spatial rate-control specifies arbitrary bitrates at different spacial locations across each frame. Traditional rate-control assigns the bitrate by varying the quantization parameter. Their rate-control process is inspired

by the Lagrangian optimization which assigns bitrates based on an estimation of the rate-distortion curve slope. They estimate the slope of the local RD-curve for each spatial location and rate. Then, they choose the largest rate such that the slope is superior or equal to their defined threshold specified during the encoding. With an NVIDIA Tesla V100, their combined methods encode and decode 640×480 videos at 2 FPS and 10 FPS, respectively.

Agustsson *et al.* [79] designed a scale-space flow for end-to-end video coding. A scale-space flow is a generalization of optical flow that adds a scale parameter to allow the network to better model uncertainty. The I frame is encoded and decoded through a specific hierarchical auto encoder. Then, P frames are processed by a scale-space flow encoder to jointly estimate and encode the quantized scale-space warp latents. To estimate the current frame, a scale-space wrapping is applied to the scale-space flow and the previous reconstruction. The scale-space wrapping starts with the construction of a fixed-resolution scale-space volume on the computed P frame. Then, a trilinearly sample is outputed from the previous 3 dimension scale-space volume using a displacement field plus a scale field which gives an estimation of the current frame. The residues obtained by the difference between the estimation and the current frame are quantized and encoded. The quantization and entropy estimation approaches are a combination of both methods [76] and [80].

These end-to-end techniques are performed without the RDO process which removes the exhaustive search made with conventional standards. The complexity issue is then transferred to the optimization of the neural networks. Furthermore, the emulation regarding end-to-end coding frameworks bring the challenge on performance reproducibility under the same conditions. In this context, Begaint *et al.* [81] proposed a PyTorch library and evaluation platform for end-to-end image and video coding methods. They provided custom operations and layers to create customized end-to-end coding techniques. This framework allows a fair comparison between traditional coding standards and state of the art end-to-end compression techniques as they provide several pre-trained models.

3.7.2 Video coding tools exploiting Machine Learning

End-to-end video coding frameworks are remodelling the global scheme of video compression. Another field of research focuses on improving the tools of traditional video coding standard with machine learning methods.

3.7.2.1 Intra prediction mode

Li *et al.* [82] proposed a deep learning method for intra prediction in HEVC. The deep learning method learns how to generate intra prediction instead of computing the exhaustive search of the traditional intra modes. Indeed, a fully connected network is trained to learn an end-to-end mapping from the reconstructed pixels to the current pixels. Instead of the intra mode prediction in HEVC, the network is taking advantage of multiple reference lines and therefore more contextual information. The network outputs directly the predicted block which is then used to calculate the residues. Eight networks are trained in order to manage the several transform unit size from 4×4 to 32×32 combined with the dual training. The dual training separates the training of the network when the intra angular modes are selected as best or when the DC or planar mode are selected as best. This dual training results in two networks per transform unit size. The deep learning method cooperates with the HEVC intra prediction process resulting in 3.4% BD-BR savings for more than 9000% encoding time increase. A lighter model is proposed with 2.8% BD-BR savings for 500% encoding time increase.

Santamaria *et al.* [83] explored linear models for intra prediction in VVC. A four layers neural network is trained to form the base approach of this method. K-modes are created with the neural network. Indeed, the three first layers are defined in the network as common layers and the final layer is a mode-specific layer. Two linear models are defined based on this neural network. The first one is a linear model which is a simplification of the neural network without the non-linearity function. For each K-modes, a matrix is generated from the weights of the neural network which are normalized row-wise. The second one is a linear model with interception to add flexibility to the computation of prediction samples. The intercept term of the linear model relies on the bias of each neural network layer. With $K = 35$ modes, the linear model brings 0.72% BD-BR savings for an encoding time of 245% compared with the VTM. The second model with interception brings 1.5% BD-BR savings for approximately the same encoding time.

3.7.2.2 Inter prediction mode

Such as intra prediction, inter mode prediction has also been studied with machine learning for quality enhancement. Indeed, Wang *et al.* [84] designed a neural network architecture for post-processing inter prediction to better use the spatial and temporal information. The network is composed of a fully connected network and a CNN. First the fully connected network is filled in with the spatial neighbouring pixels and the temporal neighbouring pixels. It outputs a matrix of the size of the current block computed. Then, this matrix is added to the prediction made by the inter mode and given as input to the CNN. The CNN outputs the residues that are finally summed with the inter prediction to improve the prediction of the current block. This method is integrated in HEVC after the traditional inter prediction when inter, merge, or skip mode are applied. It enables 1.7% BD-BR savings but for a high encoding complexity increase estimated to 3444% compared with the HM.

Benjak *et al.* [85] generated an artificial reference picture with a recurrent neural network. In the VTM with the integration of this method, after the encoding of a picture, the last coded picture is added to a ring buffer with ten entries. The recurrent neural network architecture is the one proposed in [86]. The entry of the network is the ring buffer and the output is the artificial reference picture. This new artificial reference picture replaces in the reference picture list the picture with the highest temporal difference to the current picture. This method leads to an encoding time increase of 105% for 0.94% and 0.25% BD-BR savings on the KITTI dataset [87] for straight and curve classes, respectively.

3.7.2.3 In-loop filter

Bouaafia *et al.* [88] replaced the traditional in-loop filtering by a deep learning technique in VVC. Firstly, they designed a control flag at CTU and frame level to disable the filtering if the enhancement quality is not worth the signalling cost. The RD-cost is calculated first at the CTU size and second at the frame size with and without the filtering. If the RD-cost before filtering is lower than after at the frame size, the CTU filtering flags are not transmitted due to the deactivation of the frame filtering flag. Secondly, the filtering process is performed through a wide-activated squeeze-and-excitation deep CNN with the three components YUV simultaneously. Six channels are given as input for the CNN with the three components independently, the QP, and the luminance and chrominance tree partitioning. Each input is normalized, the three components, the QP $\in [0, 1]$, and the tree partitioning $\in [1, 2]$. A partitioning map is created for luminance and chrominance partitioning information, the boundary positions are defined as 2 and the other positions

as 1. The CNN outputs the three filtered components separately. With this method, they obtain 4.5% BD-BR savings in RA configuration.

Instead of replacing the in-loop filtering, Zhang *et al.* [89] proposed to extend the filtering with a CNN for VVC post-processing. A generative adversarial network is performed after the decoding by enhancing the quality of 96×96 patch from the video. The discriminator is trained to predict how much the quality of the reference block is perceptually better than the fake one. The generator is trained using the multi-scale-Structural SIMilarity (SSIM) as a loss, then, the generator and discriminator are trained together with custom perceptual loss. Another training is proposed with the mean absolute difference only on the generator. The first custom perceptual loss training gives 0.9% and 13.9% savings in BD-BR for the PSNR and VMAF metrics, respectively. Instead, the l1 loss training performs better on the PSNR metric but poorly on the VMAF metric compared with the previous loss with 3.9% and 4.2% BD-BR savings, respectively. The proposed method is also applied on the AV1 software encoder, where it shows similar results.

3.7.3 Super-resolution pre and post-processing

Video coding introduced a new hypothesis with the evolution of the super-resolution field in order to address the challenges of network bandwidth and storage size for instance. Ho *et al.* [90] proposed a super-resolution method performed as pre-processing and post-processing on HEVC. First, a bi-cubic interpolation is performed to downsample the sequence at the scale of two. Then, the sequence is encoded using the HEVC standard. Finally a super-resolution network is applied on the decoded low-resolution sequence. Instead of transforming the decoded low resolution sequence to the high resolution sequence directly, they added a transitional state with the low-resolution sequence between the two previous states. Indeed, the super-resolution network is composed of a first restoration network that takes as input the decoded low resolution sequence and outputs a restore low resolution sequence to removes the coding artefacts from the decoded low resolution sequence. Then, a reconstruction network is applied on the restored low resolution sequence to predict the high resolution sequence. Skip connections are included between the two networks to provide the captured features from restoration to reconstruction. This solution reaches a BD-BR savings of 6.14%, 7.43%, and 10.67% for RA, LDP, and AI configurations, respectively. The complexity is decreased for HEVC as the video encoded is down-sampled, however the overall method with the downsampling and the super-resolution methods add complexity with its 1.78 million parameters.

Deuxième partie

Contributions

CHAPITRE 4

Complexity reduction opportunities in the practical Versatile Video Coding encoder

As shown in the Chapter 3, complexity reduction is a hot topic for video coding since the release of High Efficiency Video Coding (HEVC). Versatile Video Coding (VVC) brings impressive coding quality improvement compared with HEVC but at the cost of a high encoding complexity increase. Indeed, through the VVC standardization, Joint Video Expert Team (JVET) has investigated many new coding tools such as the Multi-Type Tree (MTT) partitioning, new intra and inter prediction tools, and new transforms.

As presented earlier, VVC extended the Quad-Tree (QT) block partitioning scheme of HEVC by adding the nested recursive MTT partitioning with additional Coding Unit (CU) shapes. This new tree partitioning scheme forms the basis of the VVC encoding process. VVC allows five different splits through the MTT partitioning with QT, Binary-Tree (BT), and Ternary-Tree (TT). BT and TT splits include both horizontal or vertical partitioning. Moreover, the number of intra prediction modes is extended from 35 in HEVC to 67 in VVC to better leverage spatial redundancy of the reconstructed neighbouring blocks. Indeed, angular modes in VVC are extended to 65. As VVC allows rectangular shape for a CU, wide-angle modes replace some angular modes according to the dimension (width / height) of the CU [91]. Increasing the number of intra modes provides, on average, 0.51% Bjøntegaard Delta Bit Rate (BD-BR) saving for 108% encoding complexity increase in All Intra (AI) configuration [92]. VVC introduces the Multiple Transform Selection (MTS) process that evaluates different core transforms and selects the one that minimizes the rate-distortion cost. In addition to the Discrete Cosine Transform (DCT)-II supported in HEVC, two new transforms, DCT-VIII and Discrete Sine Transform (DST)-VII, are included in the MTS process. For luma component, the MTS process is computed on CU of size equal or lower than 32×32, otherwise DCT-II is selected. For chroma component, only DCT-II is considered. In VVC Test Model (VTM)3.0, MTS provides a BD-BR saving of 2.8% for an encoding complexity increase of 238% in AI configuration [92]. Considering the aforementioned complexity increase of VVC, encoding complexity reduction is an active research field.

In this context, we propose a hierarchical characterization and evaluation of the VVC encoding complexity with the VTM encoder. We first present an overview impact of encoding parameters on the encoding complexity. The studied parameters include spatial resolution, Quantization Parameter (QP), and the split impact. Then, we define and evaluate the complexity reduction opportunities offered by three algorithmic encoding levels :

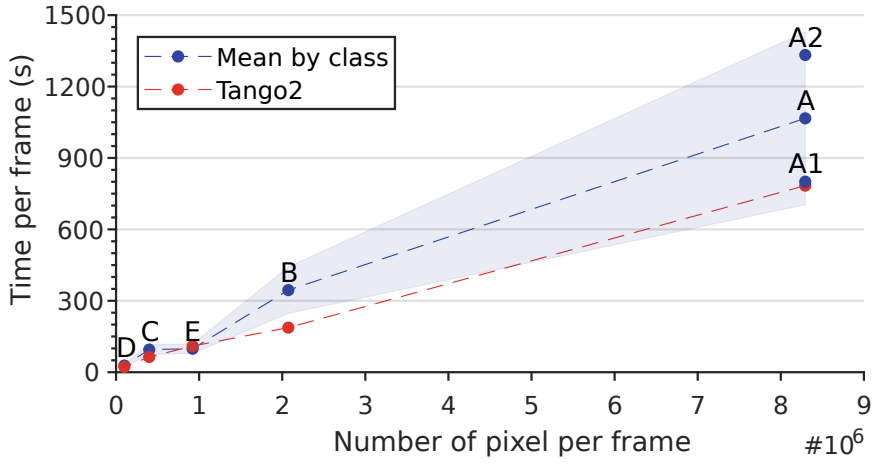


Figure 4.1 – *Average encoding time per frame in seconds as a function of the number of pixels per frame for each class and Tango2 downscaled encodings.*

the Coding Tree Unit (CTU) partitioning, the intra mode prediction, and the MTS process under the VTM3.0 in AI configuration and the CTU partitioning under the VTM10.2 for both AI and Random Access (RA) configurations. The remainder of this chapter is organized as follow. Section 4.1 details the experimental setup and evaluates the encoding complexity with different encoding parameters. Section 4.2 presents the identified complexity reduction opportunities and analyses their impacts on the VVC encoding process. Finally, Section 4.3 concludes the chapter.

4.1 Complexity analysis of the Versatile Video Coding test model encoder

Complexity reduction is a key aspect to obtain a real-time encoder. Complementary to the state of the art complexity reduction techniques introduced before, this analysis characterizes and evaluates the complexity of the VTM encoder through different parameters. The parameters assessed are the video resolution, the QP, and the split impact. The split impact is divided in two parameters with firstly the global suppression of a defined split and secondly, the suppression of certain depths of a defined split.

Our experiments related to the resolution and the QP are performed on the VTM3.0 under AI coding configuration. Whereas, our experiments related to the splits are computed on the VTM10.2 under AI and RA configurations. The sequences selected for our experiments are the JVET Common Test Conditions (CTC) [19]. All tests are carried out sequentially on Intel Xeon E5-2603 v4 processors running the Ubuntu 16.04.5 operating system. The following sub-sections examine the impact of the considered encoding parameters.

4.1.1 Impact of resolution on complexity

The VTM encoder applies the same coding process on each CTU, so its complexity is directly related to the number of CTUs and consequently to the spatial resolution and frame rate of the encoded sequence.

Figure 4.1 displays the average encoding time per frame in seconds according to the number of pixels per frame. The average encoding time per frame is plotted in blue for the

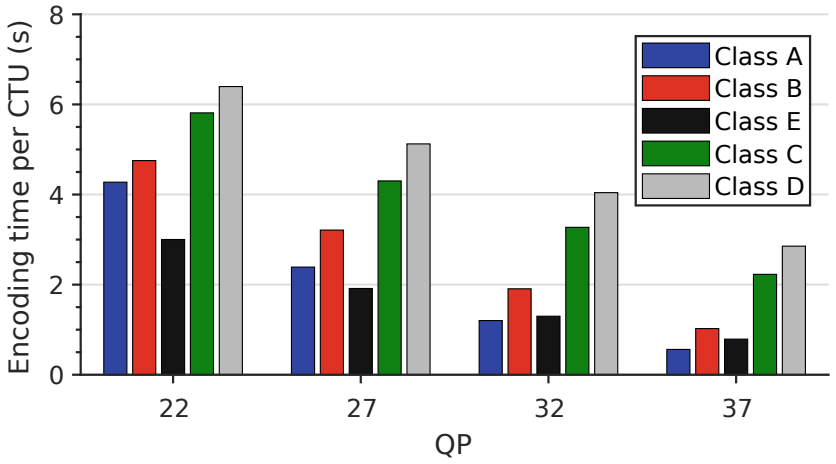


Figure 4.2 – *Average encoding time per CTU as a function of the QP, classes A-E.*

different CTC classes. Blue area represents the standard deviation of the results across the different sequences of the same class. In addition, the red line shows the average encoding time per frame for the 4K *Tango2* sequence which was downscaled by the FFmpeg bilinear filter in all CTC resolutions. *Tango2* sequence was chosen since its average encoding time per frame was close to the average of its class A1.

Figure 4.1 shows that the complexity of the VTM encoder increases with the number of pixels per frame except for class E, where the sequences have specific screen content properties such as fixed unified background which explains the irregularity. The encoding time increases linearly with the number of pixels for the same video content, as shown with the downscaled *Tango2* sequences (in red in Figure 4.1). However, the standard deviation of the mean varies between 19% and 34%, which highlights that the encoding complexity is also highly linked to the video content.

The VTM encoder includes content-dependent early termination mechanisms in the Rate-Distortion Optimization (RDO) process. One of these methods compares the Rate-Distortion (RD)-cost of the unsplit CU with the accumulated RD-cost of sub-CUs to early terminate the tree partitioning process when the accumulated cost becomes higher than the unsplit CU cost. These complexity reduction methods explain the non-stability of encoding complexity across different sequences of different video contents. High complexity difference in the same resolution is further shown by the two sub-classes A. Indeed, a ratio on the average time per frame of 1.66 between the class A1 and A2 is depicted.

4.1.2 Impact of Quantization Parameter on complexity

The QP is used both in the RD-cost computation and in the quantization step of the encoding process. Figure 4.2 illustrates the average encoding time per CTU for each class as a function of QP. The results show that the encoding complexity decreases as QP increases for all classes. For instance, with class B, the encoding time per CTU is 4.75s at QP 22 and 1.02s at QP 37. This is mainly due to the fact that an encoding with higher QP quantizes data more aggressively, leading to a larger number of zero coefficients after quantization.

The results of Figure 4.2 also show that the complexity reduction due to QP does not behave consistently among the classes. The ratio of time per CTU between QP 22 and QP 37, is equal to 7.61 for class A and 2.24 for class D. The higher the resolution, the greater

Table 4.1 – *Description of the five configurations proposed to show the impact of the split in the tree partitioning process relying of the maximal depth allowed.*

	Configuration				
Max depth	QTBTTT	QTBT	QTTT	QT	QT′
$d_{\max}^{QT}$	4	4	4	4	5
$d_{\max}^{BT}$	3	3	0	0	0
$d_{\max}^{TT}$	3	0	3	0	0
$d_{\max}^{MTT}$	3	3	3	0	0

the impact of QP on encoding time per CTU. This can be explained by the complexity reduction techniques mentioned in the Section 2.6. Indeed, according to Equation 2.12, when a sequence is encoded with high QP, more weight is assigned to the rate which leads the encoder to select larger blocks. The early termination methods included in VTM are more likely to stop the tree partitioning process earlier for high QP. This tends to further reduce the complexity. This observation also explains why the encoding time per CTU is lower for class E. These sequences have fixed unified background which leads the encoder to select larger blocks.

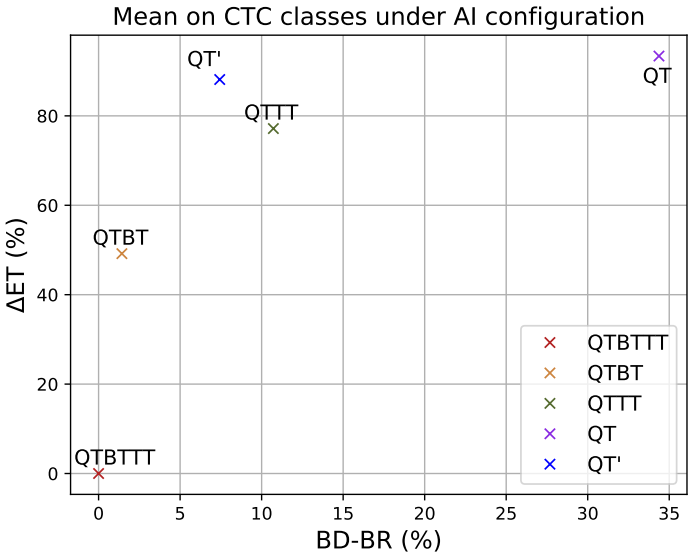
4.1.3 Impact of tree partitioning on the encoder complexity

An analysis on the splits is deduced for the VTM10.2 under AI and RA configurations. This analysis shows the contribution of the new adopted splits BT and TT in VVC by deactivating them independently. The five configurations QTBTTT, QTBT, QTTT, QT, and QT' defined in TABLE 4.1 are tested. This table represents the five configurations proposed alongside the maximal depth of QT, BT, TT, and MTT allowed defined by $d_{\max}^{QT}$, $d_{\max}^{BT}$, $d_{\max}^{TT}$, and $d_{\max}^{MTT}$, respectively. The first configuration QTBTTT is the reference relying on the classical VTM without any modification. The other configurations QTBT, QTTT, and QT are the VTM without BT, TT, or both splits. Finally, the last configuration QT' represents the VTM computed without both BT and TT splits but with an increase of the $d_{\max}^{QT}$ to five which is the closer configuration to HEVC in terms of tree partitioning process.

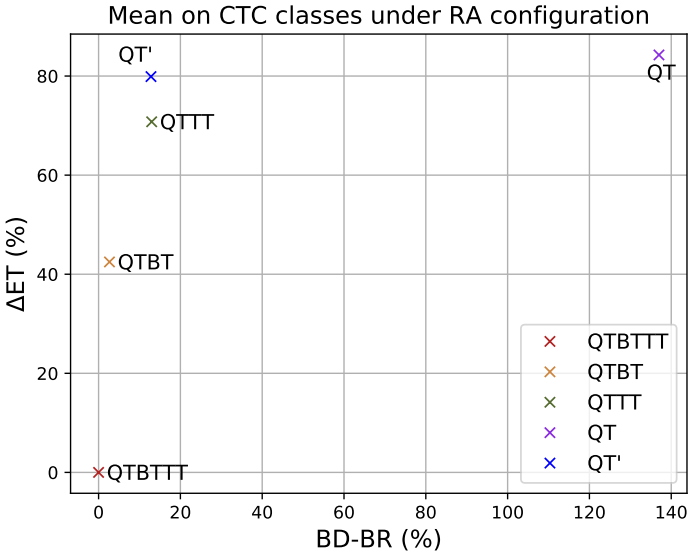
The complexity reduction opportunities are defined by ΔET computed in Eq. 2.7 (page 16). The reference complexity and BD-BR values are the classical VTM with QT, BT and TT splits enabled which is defined by the QTBTTT configuration.

Figure 4.3a detailed the ΔET which is the complexity reduction and the BD-BR loss versus the classical VTM encoder in AI configuration. Disabling TT reduces the complexity by half for a BD-BR loss of 1.43%, while disabling BT reduces the complexity by 77.2% for a high BD-BR loss of more than 10%. When BT split is deactivated, it brings a higher complexity reduction but at the expense of higher BD-BR loss compared to TT. The importance of BT is way above the TT split for the coding gain when taken alone. Using only QT split brings 93.4% complexity reduction for a high BD-BR loss of 34.37%. It is interesting to observe that QT extended with maximum depth has less BD-BR loss and more complexity reduction than QTTT. Indeed, when only QT is available through the depths less limitations are applied than the QTTT configuration. For instance, when the TT split is selected, QT is not allowed anymore. Both newly adopted BT and TT splits combined are crucial for the VVC coding quality under AI configuration. Indeed, the coding gain brought by the other tools are amplified by the new splits as it brings more flexibility on the blocks.

FIGURE 4.3 – *Impact of the splits relying on the configurations defined in TABLE 4.1 on the VTM10.2.*



(a) *AI configuration.*



(b) *RA configuration.*

Table 4.2 – *Description of the seven configurations proposed to show the impact of the split depth in the tree partitioning process relying of the maximal depth allowed.*

	Configuration						
Max depth	QTBTTT	MTTD2	MTTD1	BTD2	BTD1	TTD2	TTD1
$d_{\max}^{QT}$	4	4	4	4	4	4	4
$d_{\max}^{BT}$	3	2	1	2	1	3	3
$d_{\max}^{TT}$	3	2	1	3	3	2	1
$d_{\max}^{MTT}$	3	2	1	3	3	3	3

Compared with the previous presented figure, Figure 4.3b analyses the splits under the RA configuration. Same as for AI configuration, BT brings more impact to the coding quality with almost 13% BD-BR loss when deactivated with 70.8% complexity reduction while TT introduces 2.66% BD-BR loss for 42.5% complexity reduction. When both splits are deactivated and only QT can be used to perform the RDO, the complexity reduction is up to 84.3% for 137% BD-BR loss. Same as under the AI configuration, it is due to the limitations on the VTM that most of the time stop QT at block of size 16×16.

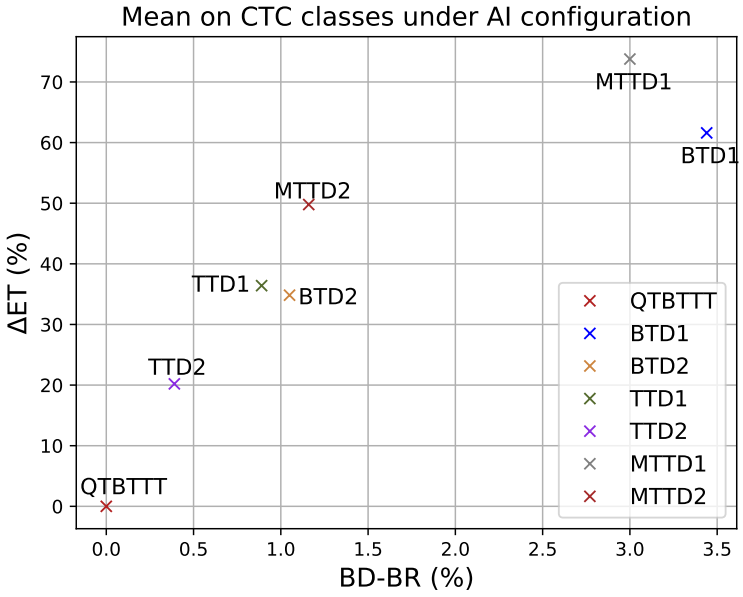
The impact on complexity and coding loss is different between AI and RA configurations as the tree partitioning is more relying on the spatial or temporal information, respectively. However, BT has a greatest impact on the coding quality than TT for both configurations as it is less specific with its division in two sub-blocks. When using only QT, the BD-BR loss is tremendous especially under RA configuration. Nevertheless, it is not only due to the missing of BT and TT splits but also to the depth limitation. QT split is performed until the block size is superior to 8×8 plus it is less adaptive to the content with its division in four sub-blocks of the same size. Furthermore, due to the VTM early termination, most of the blocks are stopped at size 16×16 such as Advanced Video Coding (AVC). When the QT depth is extended to the maximal, the QT only has better results than QT limited in depth plus TT. The comparison between AI and RA configurations shows that the splits have higher gain on BD-BR under RA. The temporal coding brings more opportunity to reduce the BD-BR which is amplified by the possibilities brought by the new BT and TT splits.

4.1.4 Impact of split depth on the encoder complexity

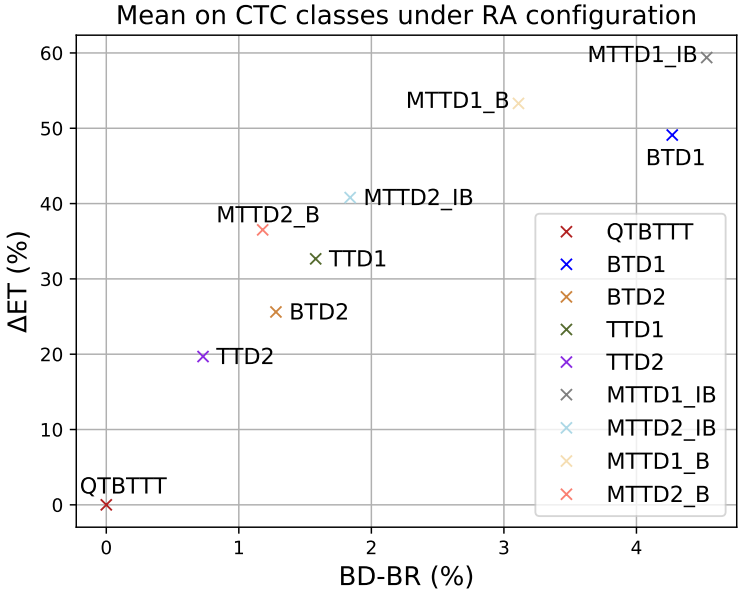
Not only the splits are studied but also the impact of the depths for each split. This study is also performed under AI and RA configurations for a complete analysis with the VTM10.2. Such as previously, the reference VTM is used as comparison with the classical QT, BT and TT splits. Furthermore, the classical VTM sets the MTT maximal depth to 3.

This analysis determines the complexity reduction and BD-BR loss based on different configurations detailed in TABLE 4.2. Those configurations modify the maximal depth of MTT, BT, and TT split independently. Implicit BT split can occurs when the frame size is not a multiple of the CTU size. However, the implicit BT split is not added to the MTT current depth. Therefore, in order to skip the BT and TT depths, the difference between MTT depth and the implicit MTT depth needs to be higher or equal to the selected depth $MTT_{depth} - implMTT_{depth} \geq depth$. This enables to skip the split when the current MTT depth is superior or equal to the selected depth. To skip the global MTT depth, the desired depth is set on the configuration file of the VTM. These analyses could also be considered as basic complexity reduction techniques or minimal complexity reduction BD-BR trade-off by managing the depth across the splits.

FIGURE 4.4 – *Impact of the splits relying on the configurations defined in TABLE 4.2 that modify the maximal split depth for MTT, BT, and TT independently on the VTM10.2.*



(a) *AI configuration.*



(b) *RA configuration.*

Figure 4.4a presents the different split depth configurations with on the x-axis ΔET and on the y-axis the BD-BR loss under AI configuration. As presented before, the TT split brings less complexity reduction for a lower BD-BR loss compared with BT. The configuration TTD2 confirms the previous results, the complexity reduction is about 20% for 0.39% BD-BR loss. It is more interesting to skip the TT split at depth 1 (configuration TTD1) than the BT split at depth 2 (configuration BTD2) in terms of trade-off between complexity reduction and BD-BR loss. The BTD1 configuration reaches 61.6% complexity reduction for 3.44% BD-BR loss. Finally, when skipping MTT at depth 2 (configuration MTTD2), it halves the complexity for 1.16% BD-BR loss. For MTTD1, it brings higher reduction of the complexity with 73.8% which is the highest reduction, but for 3% BD-BR loss.

Figure 4.4b presents the same results as Figure 4.4a but targeting RA coding configuration. New configurations are detailed in this figure with MTT_B and MTT_IB which means that the split depths are limited only for B frames or through the whole sequence across I and B frames. The other configurations illustrated in this figure are skipping the depths for I and B frames. Such as under AI configuration, TT has lower complexity reduction and lower BD-BR loss. However, the configuration BTD2 is interesting under RA configuration compared with AI as it achieved a lower BD-BR loss than TTD1. BTD1 is reducing the complexity by 49.1% for a BD-BR loss of 4.27%, compared with AI, it is 12% complexity reduction lower and almost a BD-BR loss higher by 1%. Splits have more impact on the BD-BR in RA configuration than in AI for lower complexity reduction. As shown in the figure, the MTT depth restrained only on B frames has lower complexity reduction by approximately 5% compared with MTT restrained on I and B frames with both depth restricted to 2 and 1. Furthermore, with the same depth comparison, i.e., MTTD1_B versus MTTD1_IB and MTTD2_B versus MTTD2_IB, the BD-BR loss is increased by 0.6% and 1.5%, respectively. Indeed, the first I frame has an high impact on the BD-BR since it is used as a reference for other frames through temporal interpolations.

To conclude, this analysis shows that the complexity of video encoding is related to the frame rate, resolution, and QP value. However, our results also bring out that the encoding complexity is content dependent. Moreover, the splits have an important impact on the BD-BR and the complexity as they influence the tools further applied in the encoder. Next, we analyse the complexity reduction opportunities at the coding tool level.

4.2 Complexity reduction opportunity in the Versatile Video Coding test model encoder

Complexity reduction techniques commonly reduce the number of tested configurations while trying to limit the degradation of the RD-cost. In the VTM encoder, an exhaustive RDO search that leads to the minimal RD-cost is performed at three nested levels : 1) CTU partitioning ; 2) intra mode prediction ; and 3) MTS process. The next section determines the complexity reduction opportunities at these three levels.

4.2.1 Determination of the complexity reduction opportunities

A theoretical upper limit for complexity reduction is obtained when the encoder is able to predict perfectly the best configuration and thus only this configuration is processed. Therefore, for a given level, the complexity of the search process is reduced to the minimal complexity for exactly the same coding performance. This encoding complexity sets

Table 4.3 – *Analysed configurations with maximum available complexity reduction.*

	Configuration							
Level	Ref	C_0	C_1	C_2	C_3	C_4	C_5	C_6
CTU partitioning	E	D	E	E	D	D	E	D
Intra mode prediction	E	E	D	E	D	E	D	D
MTS process	E	E	E	D	E	D	D	D
ΔET (%)	0	97.48	65.20	55.16	98.66	98.45	82.21	98.97

D : Disabled, E : Enabled.

the theoretical complexity reduction opportunities of the according level, i.e., the CTU partitioning, the intra mode prediction, and the MTS process.

Let the *optimal configurations* of CTU partitioning, intra mode prediction, and MTS process be the ones with the minimum RD-cost. These configurations are determined with a two-pass encoding. The first pass is unconstrained, i.e., an exhaustive RDO search is processed, and the optimal configurations of the three levels are extracted. The second pass uses the extracted optimal configurations to force the RDO process to encode only these configurations and thus to remove unnecessary complexity. The encoding results of these two passes are identical in terms of bit rate and distortion, only the complexity is different.

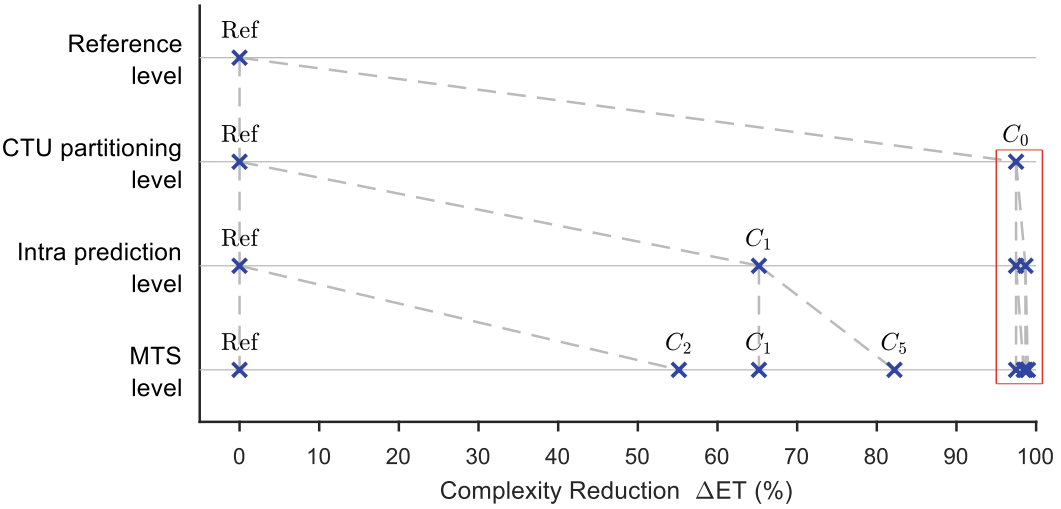
4.2.2 Analysis of complexity reduction opportunities for Versatile Video Coding test model 3.0 under All Intra configuration

Table 4.3 presents seven analysed configurations (C_0-C_6) which feature different combinations of the three levels defined in Section 4.2.1. For each level, the exhaustive search is either enabled (E) or disabled (D). When the exhaustive search is disabled at one level, only the optimal configuration of this level is performed. Following Equation 2.7, Ref represents the reference encoding complexity (exhaustive search) that is used as reference encoding time T_R and T_C the encoding time forced to encode the optimal configuration represented as C_i to compute the complexity reduction ΔET. C_0, C_1, and C_2 represent the complexity reduction opportunities of the CTU partitioning, intra mode prediction and MTS levels, respectively. The other configurations are combinations of the previous ones.

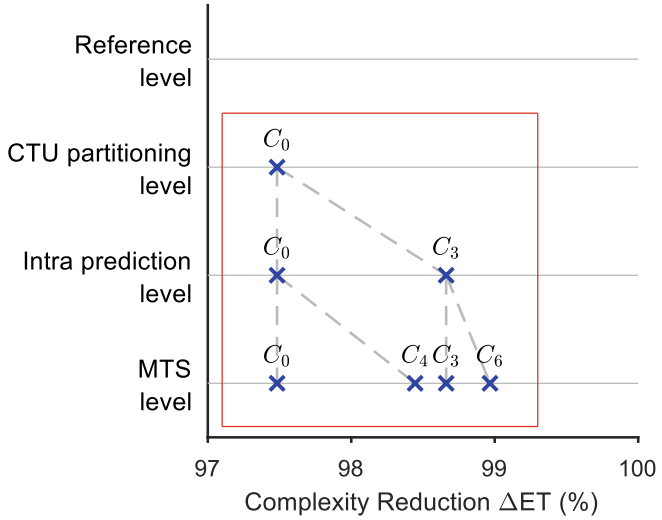
Figure 4.5a shows complexity reduction of these seven configurations organized at the three nested levels. The complexity reduction results are averaged for the 22 CTC sequences. The reference (Ref) is used as an anchor. As explained in the previous section, all configurations produce identical bit-rate and distortion, i.e, BD-BR differences between the reference and the other configurations are equal to 0. Figure 4.5b focuses on the results between 97% and 100% of Figure 4.5a as the results are too close to be distinguished. The average standard deviation of the results presented in Figure 4.5a is equal to 2.11% (with a maximum of 4.89%) which confirms that the average results across QPs and classes are representative. To interpret the following results, it is important to note that the experiments are performed under the VVC reference software VTM3.0, which is not implemented to be a practical real-time encoder.

As shown by the C_0 configuration on Figure 4.5a, restraining the encoder to process only the optimal configuration for the CTU partitioning level offers the best complexity reduction opportunity among the three levels, up to 97.5%. In other words, being able to

Figure 4.5 – *Maximum complexity reduction (ΔET) of the analysed configurations over the default configuration of VTM3.0.*



(a) *Whole range from 0% to 100%.*



(b) *Zoomed range from 97% to 100%.*

perfectly predict the CTU partitioning without testing the unnecessary splits would reduce the encoding time down to 2.5% of the reference. This result is due to the multitude of split possibilities introduced in the VTM, including QT, BT, and TT. The RDO process, including the search of intra mode and transform, is applied for each CU. Reducing the number of intra modes tested by the RDO process for all blocks can reduce the complexity by 65.2% over the reference with 67 modes, as presented by the C_1 configuration on Figure 4.5a. Finally, the MTS process enables a complexity reduction opportunity of 55.2% when the optimal horizontal and vertical transforms can be predicted for all blocks. The MTS in the VTM tests several transforms for each block which significantly increases the encoding time.

Since the aforementioned three levels are nested but independent, it is possible to process only the optimal configuration of multiple levels simultaneously. Reducing the complexity of multiple levels is interesting as it was previously done on several works for HEVC [33, 34]. As shown by the C_5 configuration on Figure 4.5a, addressing the intra mode prediction and MTS together offers a complexity reduction opportunity of 82.2%. This result is in line with the individual complexity reduction opportunities of the intra prediction and MTS level ($65.2\%+(100\%-65.2\%)\times 55.2\% = 84.4\% \approx 82.2\%$). As shown in Figure 4.5b with the C_3, C_4, and C_6 configurations, when intra prediction and MTS levels are combined with the CTU partitioning level, the complexity reduction opportunities are not much higher than that of the CTU partitioning level alone (less than 2% of difference). Indeed, if the CTU partitioning is predicted perfectly, the complexity used by the RDO process to select the intra prediction mode and transform is very low as the RDO process is only done once on the selected blocks.

Considering the current version of VTM3.0, we can conclude that the complexity reduction issue can be more efficiently addressed by optimizing the CTU partitioning process rather than the intra mode prediction or MTS process. Furthermore, knowing the theoretical upper limits of complexity reduction at different levels of VVC may help to better evaluate current and future optimization techniques. To illustrate that, a method cutting 50% of the VTM complexity through the MTS process has almost reached its maximum whereas the same result on the CTU partitioning is only half of the theoretical maximum.

4.2.3 Complexity reduction opportunities for the Versatile Video Coding test model 10.2 under All Intra and Random Access configurations

Such as before, we proposed to establish the maximum complexity reduction opportunities for the current VTM version 10.2. Unlike the analyses on the VTM3.0, this part focuses only on the tree partitioning algorithm and its possible acceleration by early skipping unlikely splits. This allows to point the consistency on the complexity reduction opportunity between two VTM versions. Indeed, the tree partitioning is a high level process that influences almost all the tools inside the VTM encoder. Furthermore, if the tree partitioning process is limited, it also directly avoids the other tools such as prediction, transform, or quantization for the blocks removed by the non-optimal splits.

TABLE 4.4 presents the complexity reduction opportunity of the tree partitioning relying on the VTM10.2 under AI and RA configurations. Such as described before, the complexity reduction opportunity under AI configuration is almost the same with 94%. This small decrease is certainly due to the new tools proposed in this recent version that add complexity plus the in-loop filters that are out of the scope of the tree partitioning process. All CTC classes have more than 90% complexity reduction opportunity for the tree parti-

Table 4.4 – *Complexity reduction opportunities for the tree partitioning process with the VTM10.2 under AI and RA configurations through the CTC.*

Class	Sequence	AI configuration	RA configuration
		ΔET	ΔET
Class A1	Tango2	86.5%	92.8%
	FoodMarket4	89.5%	89.5%
	Campfire	94.9%	96.6%
	Average	**90.3%**	**93%**
Class A2	CatRobot1	93.8%	92.6%
	DaylightRoad2	94.6%	92.9%
	ParkRunning3	96.2%	94.9%
	Average	**94.9%**	**93.5%**
Class B	MarketPlace	95.5%	92.7%
	RitualDance	94.1%	93.6%
	Cactus	96%	94%
	BasketballDrive	94.9%	94.3%
	BQTerrace	95.8%	90.9%
	Average	**95.2%**	**93.1%**
Class C	RaceHorses	95.9%	95.6%
	BQMall	95.5%	92.9%
	PartyScene	96.3%	93.2%
	BasketballDrill	95.1%	93.1%
	Average	**95.7%**	**93.7%**
Class D	RaceHorses	94.2%	92.5%
	BQSquare	94.1%	79.1%
	BlowingBubbles	94.7%	90.2%
	BasketballPass	92.9%	87%
	Average	**94%**	**87.2%**
Class E	FourPeople	95%	87.7%
	Johnny	92.9%	83.3%
	KristenAndSara	93.4%	86.5%
	Average	**93.8%**	**85.8%**
	Average	**94.2%**	**91.2%**
	Standard deviation	2.2%	4.1%
Class F	ArenaOfValor	95.9%	93.5%
	BasketballDrillText	95.3%	93%
	SlideEditing	96.1%	83.9%
	SlideShow	91.4%	82.8%
	Average	**94.7%**	**88.3%**

tioning process with a slight standard deviation of 2.2% which indicates the homogeneity through the sequences. Under RA configuration, the complexity reduction opportunity is slightly lower for the tree partitioning process with 91.2%. For high resolution sequences, the complexity reduction opportunities remain close to the AI results. However, with lowest resolutions or specific sequences such as class F, complexity reduction opportunities under AI configuration are greater than under RA configuration. This can be due to the inter prediction tools that take an important part in the complexity. The standard deviation under RA configuration is also higher with 4.1% which shows less homogeneity under the sequences.

These results prove the consistency of the complexity reduction opportunity across the VTM versions and across the configurations. Furthermore, as the tree partitioning is an high-level algorithm, this process includes other processes that have also high complexity. By reducing the tree partitioning search, all the tools computed after this are also reduced which leads in a high complexity reduction opportunity of more than 90%.

4.3 Conclusion

Complementary to state of the art techniques, this chapter analysed the complexity reduction opportunities of the VTM encoder by using a hierarchical approach, from encoding parameters to encoding tools. This study demonstrated that the relative complexity of the VTM encoding is proportional to video resolution and QP. It also shows that the splits have an important impact on the VTM encoding complexity and BD-BR loss. TT has a low influence on the BD-BR compared with BT for a non negligible complexity overhead. Furthermore, the main contribution of this chapter was to bring out the theoretical upper limits for various coding tools. The CTU partitioning is shown to have high potential for complexity reduction with 97% whereas the respective percentages for intra mode prediction and MTS are 65% and 55%, respectively with VTM3.0 under AI configuration. Complexity reduction opportunity for the VTM10.2 has the same result than the one on VTM3.0 under AI configuration for the tree partitioning process. Under RA configuration, the complexity reduction opportunity of the tree partitioning process reaches 91.2% which is close to the AI opportunity.

The results of this study motivated our contributions, presented in the rest of this thesis, that mainly focus on the prediction of the tree partitioning in the VVC encoder.

CHAPITRE 5

Deep learning based complexity reduction of Versatile Video Coding intra encoder

5.1 Introduction

Chapter 4 has presented the analysis on the complexity of the Versatile Video Coding (VVC) Test Model (VTM) encoder. It demonstrated that the tree partitioning process has the largest complexity reduction opportunity with 97% compared with the intra mode prediction or the Multiple Transform Selection (MTS) processes under All Intra (AI) configuration. This chapter focuses on the Convolutional Neural Network (CNN) which extracts internal features to decide the split to test. This CNN will be exploited to reduce the complexity of the VTM encoder under AI configuration in the next chapters.

In this chapter the first part of the global method to reduce the complexity is presented. Indeed, a CNN is trained in order to predict features that will be used to select the optimal split. Those features are split probabilities associated all edges at each 4×4 block. The CNN structure is detailed through its input, output, and layers. The Quantization Parameter (QP) is part of the input alongside with the 68×68 luminance pixels as it has an important impact on the tree partitioning. Our image dataset is optimised in order to maximise the CNN performance which reaches 86% of good probabilities. Furthermore, with the optimisation proposed and a Graphics Processing Unit (GPU), the CNN is able to predict the tree partitioning at 16.2 Frames per Second (FPS) for a full HD sequence.

The remainder of this chapter is organized as follows. The overall method which is further detailed in the next chapters is globally presented alongside with the CNN structure in Section 5.2. Section 5.3 describes the dataset and its optimisation that is used to train our CNN. The CNN performance is presented in Section 5.4 with its loss throughout the training and several Receiver Operating Characteristic (ROC)s. Section 5.5 details the optimisation proposed to limit the CNN inference complexity on Central Processing Unit (CPU) and GPU. Finally, Section 5.6 concludes this chapter.

5.2 Intermediate features extraction with Convolutional Neural Network

The complexity reduction can be performed with binary classifier or multi-class classifier to select the optimal split among the available ones. The Binary-Tree (BT) and Ternary-

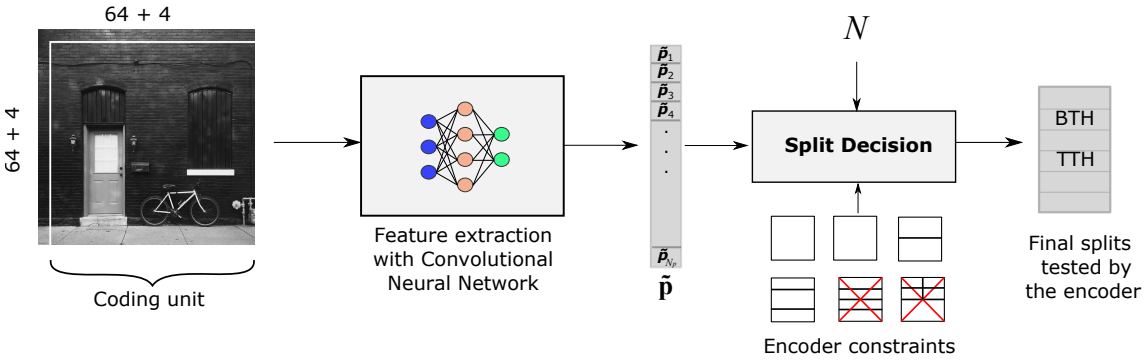


FIGURE 5.1 – *Workflow diagram of the proposed tree partitioning scheme for VVC AI coding. The input luminance block is first processed by a CNN to predict* $\tilde{\mathbf{p}}$*, a vector of* N_p *probabilities describing all edges at each* 4×4 *sub-block. The vector* $\tilde{\mathbf{p}}$ *is then processed by split decision techniques in order to determine the optimal split to process in the Rate-Distortion Optimization (RDO).*

Tree (TT) splits introduced in the VTM push machine learning techniques to multiply the number of classifiers [63], [58]. Indeed, a multi-class classifier is needed for each block size, i.e., 31 block size are available in the VTM. Furthermore, the deep learning brings more precision than Decision Tree (DT)-based techniques.

For instance, in [58], the authors proposed a cascade decision framework through 5 binary classifiers. The multiplication of classifiers can be an issue with their high computational inference time. The CNN is the basis of our complexity reduction methods which will be presented in the next chapters. However, a CNN is computationally intensive that makes it necessary to limit the number of inferences as well as the CNN structure size. The objective of our solution is to limit the access to the CNN by predicting intermediate relevant features to decide of the unlikely splits, i.e., our CNN predicts the whole 64×64 partition information to limit the number of inferences. The considered intermediate features are the probabilities of all the recursive splits within the 64×64 block. Furthermore, the number of parameters is low compared with the state of the art CNN in order to limit its complexity. Those output probabilities will be then exploited to define the unlikely splits to skip. The section presents the overall method alongside with the CNN structure.

5.2.1 Overall method

The Figure 5.1 presents the global scheme of our methods to skip unlikely splits based on the CNN output. First, the current block 64×64 combined with its 4 top rows and its 4 left columns exploited by the Multiple Reference Line (MRL) result in a 68×68 input luminance block which is processed by the CNN to predict $\tilde{\mathbf{p}}$. $\tilde{\mathbf{p}}$ is a vector of N_p probabilities describing all edges at each 4×4 sub-block. Then, two methods which will be specified in the next chapters determine the split decision process. Finally, those likely splits are processed by the VTM in order to maximize the complexity reduction while minimizing the Bjøntegaard Delta Bit Rate (BD-BR) loss.

Indeed, our method is a top-down solution that early terminates unlikely splits, i.e., it follows the hierarchical process of the VVC encoder and skips splits that have low probabilities to belong to the optimal tree partitioning. Our complexity reduction method is composed of a CNN that extracts spatial features and a split decision block that defines the unlikely splits. The features extraction block consists of a CNN that processes an input luminance block $\mathbf{B}$ to predict a vector of probabilities $\tilde{\mathbf{p}}$ of splits at the 4×4 block edges :

$$\tilde{\mathbf{p}} = f_\theta(\mathbf{B}), \tag{5.1}$$

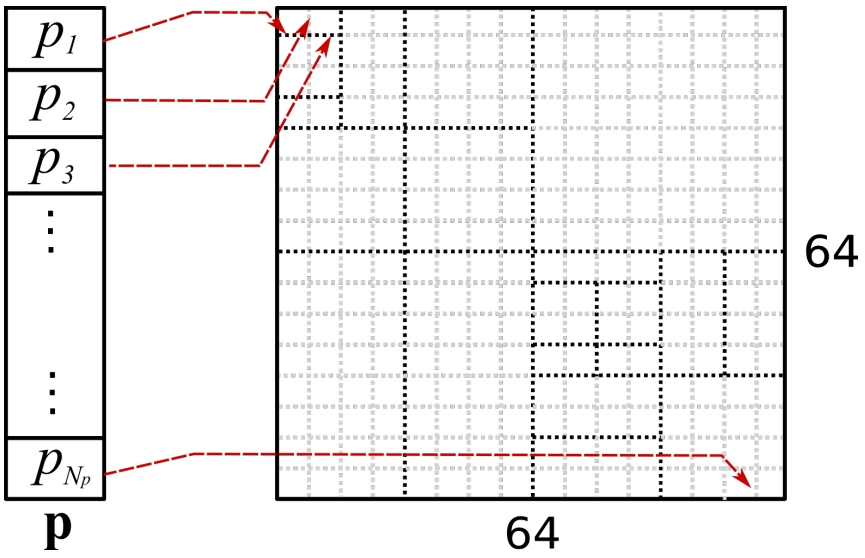


Figure 5.2 – *Correspondence between the CNN output vector and a* 64×64 *CU partition.*

where f_θ is a parametric function of the CNN with training parameters θ and $\mathbf{B}$ is the input square block of size 68×68.

The block $\mathbf{B}$ consists of the current block of size 64×64 padded with four rows and four columns on the top and left of the block resulting in a block size of 68×68. The left and top neighbour pixels are used as reference samples in the intra prediction through the MRL intra prediction tool [93]. Indeed, their pixels are used to derive the intra mode of the current block. Therefore, it is important to include these samples in the features extraction stage. The CNN predicts for each $N_B \times N_B$ Coding Unit (CU) a vector of N_p probabilities of a split at each 4×4 edge of the block. The length N_p of the predicted probabilities vector $\tilde{\mathbf{p}}$ is computed as follows :

$$N_p = \frac{N_B}{2}\left(\frac{N_B}{4} - 1\right). \tag{5.2}$$

In the case of a CU size of 64×64, N_p is the length of the vector $\tilde{\mathbf{p}}$ which is equal to 480. Figure 5.2 presents the connections between the probability vector components and the 480 4×4 edges of an input CU. As highlighted in this figure, the first $\mathbf{p}_1$ and the third $\mathbf{p}_3$ components of the vector correspond to the bottom horizontal edges of the first and second 4×4 blocks, respectively. For a good tree partitioning prediction, these two components must have a value (probability) close to 1 as a TT split is performed to split the block. The last component of the vector $\mathbf{p}_{N_p}$ has a value (probability) close to 0 since no split is performed at this last vertical edge of the CU as illustrated in Figure 5.2.

Then, the outputted vector is exploited through different techniques that will be further explained in the next chapters.

5.2.2 Convolutional Neural Network structure

Figure 5.3 presents the adopted CNN architecture which is inspired by the ResNet network [51]. The orange layers represent convolution layers with 3×3 kernel (Conv 3×3), whereas the yellow layers denote convolutions with 1×1 kernel (Conv 1×1) that transform the input feature map matrix to match the dimension of next layer which are then summed up (green plus). The red layers correspond to the max pooling (Max Pool) that subsample

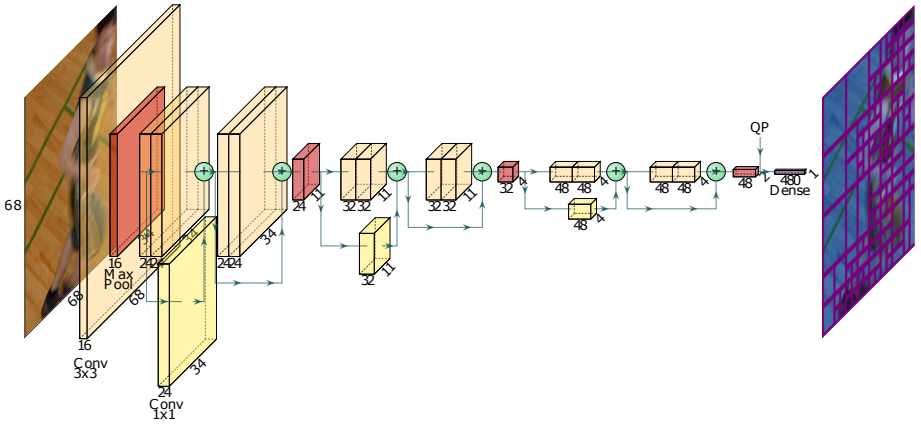


Figure 5.3 – *The CNN architecture with convolution layers in orange and yellow, max-pooling layers in red, and fully connected layer in purple.*

Table 5.1 – *Breakdown of our dataset by resolution.*

Resolution	240p	480p	720p	1080p	4K	8K	Total
Nb images	500	500	579	2557	654	418	5208

the input features map with a window of 2×2 by selecting the maximum over four values. The last layer in purple is a fully connected layer (Dense) that predicts the 480 components vector. The sigmoid activation function is used to predict the output values within the interval $[0, 1]$. All these layers result in a network with 226 088 training parameters.

The CNN input consists of a 68×68 luminance pixels of the CU currently processed plus the QP value that highly influences the final partition. The QP is provided as an external input to the last fully connected layer. The QP parameter is crucial to save the memory bandwidth as it leads to only one model shared for all QP values instead of adopting one model by QP value. Therefore, our model can be efficiently used with a rate control mechanism that adapts the QP value to the target bandwidth. The output is a 480 components vector that represents the fine-grain partitions (4×4) of the 64×64 CU. This solution has the advantage of predicting the whole CU spatial features in one shot, which is very convenient to reduce the complexity overhead and latency introduced by this step. Moreover, the architecture of the network is less deep than state of the art CNN architectures [94], [95] and thus will require less time to predict the output vector.

5.3 Convolutional Neural Network training

As presented in the previous section, the CNN is the basis of our complexity reduction methods. The training of the CNN has an high impact on the quality of the encoding. Indeed, the unlikely splits are selected related firstly to the CNN prediction. The training of the CNN is presented in this section with the optimized dataset.

5.3.1 Dataset for training

The lack of public dataset providing encoded blocks with the VTM and their corresponding partitions drives us to construct our training dataset to optimize the proposed models weights. As our work focuses on AI configuration, temporal relationship between frames is not considered. Therefore, five public image datasets were selected including Div2k [67], 4K

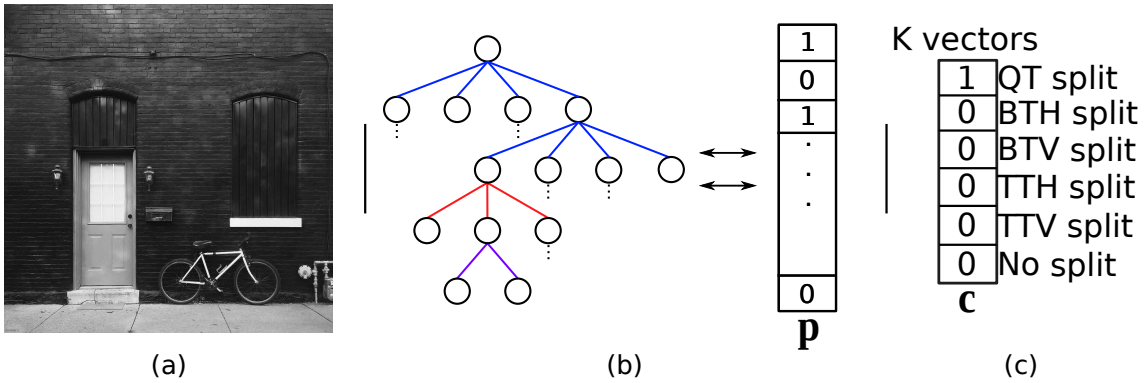


FIGURE 5.4 – *Representation of our dataset. (a) is a luminance* 68×68 *input block* $\mathbf{B}$*. (b) is the optimal tree partitioning of the block represented by a tree and transformed to the soft representation, i.e., a 480 elements vector* $\mathbf{p}$*. (c) is the hard representation,* K *vectors* $\mathbf{c}$ *of 6 probabilities, that define the optimal split for each of the* K *blocks in the tree.*

images [96], jpeg-ai [97], HDR google [98], and flickr2k [99]. The resulting dataset presents a high diversity of still image contents. However, since these datasets include more images in high resolution (Full HD and 4K resolutions), a set of high resolution images are downscaled with a bilinear filter and added to the dataset. This results in a dataset with around 5208 images at different resolutions as detailed in TABLE 5.1 which reports the number of images per resolution. The images of the same resolution are then concatenated to build a pseudo-video sequence. This latter is encoded with the VTM encoder in AI configuration with the Common Test Conditions (CTC) QP values, QPs $\in \{22, 27, 32, 37\}$.

It should be noticed that the VTM encoder includes multiple speed-up techniques for the tree partitioning process to overcome the complexity brought by this VVC process [8]. To achieve a high coding efficiency by testing more tree partitioning configurations, these speed-up techniques have been disabled to build our dataset. Compared with the VTM anchor, disabling these speed-up techniques enhances the coding efficiency with more accurate tree partitioning configurations. Nevertheless, a higher encoding time is needed to create the ground truth but only one encoding pass is required, so, increasing the encoding time is not critical at this stage. The VTM in AI configuration relies on the dual tree tool that performs separate tree partitioning for luminance and chrominance components. The tree partitioning information of both components is recorded while only the prediction of luminance tree partitioning is considered in this work since it takes the most part of the encoding complexity with more than 85% of the total encoding time [9].

The optimal partitions computed by the VTM encoder is first saved as a tree. Therefore, in order to integrate the dataset in our proposed method described in the next chapters, two data representations are defined as soft and hard representations as illustrated in Figure 5.4. Figure 5.4-(a) shows a block $\mathbf{B}$ processed by the VTM encoder which is a 64×64 luminance block plus 4 rows and columns on top and left of the block, respectively. These supplementary pixels are required for the MRL intra prediction tool. Figure 5.4-(b) illustrates the partition tree of this block which is converted to a 480 components vector. This soft representation of the tree depicts the 64×64 block luminance partition in each 4×4 edge of the block in a single vector $\mathbf{p}$. The hard representation of our dataset is a succession of K vectors that define the optimal split at each depth. Figure 5.4-(c) presents one of those vector $\mathbf{c}$ which defines the optimal split for a specific CU size. The vector size is set to six as it is the maximum number of splits defined by VVC. For instance, the 64×64 CU size has only two possible splits with Quad-Tree (QT) and no split and thus

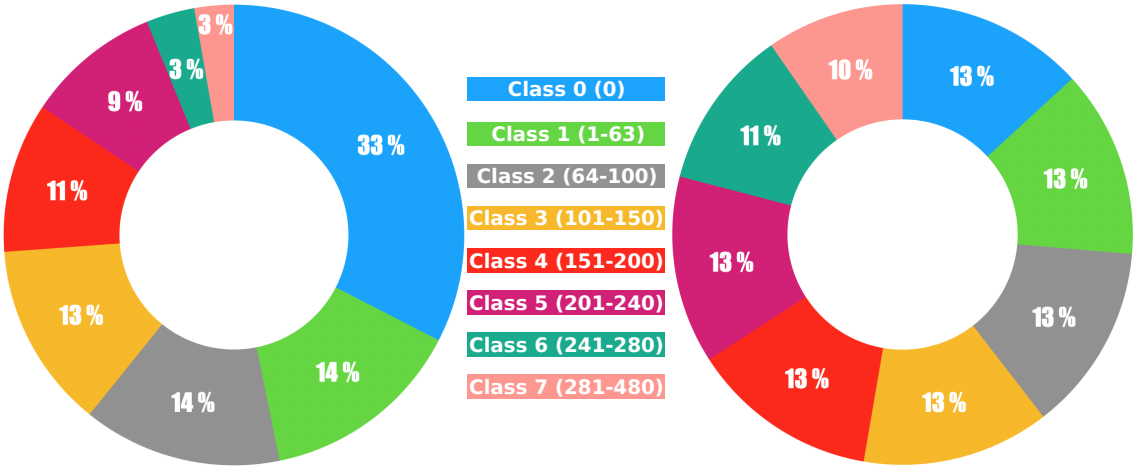


Figure 5.5 – *Breakdown of our dataset by partition classes. (left) Unbalanced dataset. (right) Balanced dataset.*

the rest of four vector components are set to zero. Instead, for a CU size of 32×32, all splits are available, so the vector size is still 6 with QT, the two BTs, the two TTs, and no split.

5.3.2 Dataset optimisation

Our dataset includes more than 26 million instances of 64×64 block partition excluding any CTC sequence to insure a fair comparison of our methods against state of the art techniques. In order to analyse the dataset disparity, a first selection of 2 million instances are picked randomly. Furthermore, to address the data heterogeneity issue of the dataset, several classes are defined with 8 levels of depth partition from no partition to deep partition. Figure 5.5 gives a percentage of appearance of each of these 8 depth levels in the dataset which are defined based on the number of edges activated (split) in the 64×64 CUs. Figure 5.5-(a) represents the class repetitions through 2 million instances before the dataset balancing. It can be noticed that the class 0 without any split contains more than a third of the 2 million 64×64 CUs instances, while the last two classes which illustrate deep partition are under-represented and must therefore be increased. Figure 5.5-(b) illustrates the depth distribution of the tree partitioning over the eight classes after the dataset balancing. 1,200,000 instances (64×64 block) were selected at each QP value with a homogeneous representation over the eight classes. The two last classes are slightly under-represented as highly deep partitions are derived by the encoder only for extremely complex blocks encoded at low QP values.

The hard representation is composed of sub-datasets separated by the different CU sizes. These sub-datasets are also balanced to enhance their representation. Indeed, we balanced individually each sub-dataset over the available splits and the QPs. For instance, the 4×16 CU size dataset is balanced between the proportion of BTH, TTH, and no split but also among the 4 QPs.

5.3.3 Training process description

The CNN is trained from scratch with the proposed dataset described previously relying on the Keras framework [100] running on top of Tensorflow module [101]. The weights θ of the CNN are updated at each batch iteration with the ADAM stochastic gradient descent

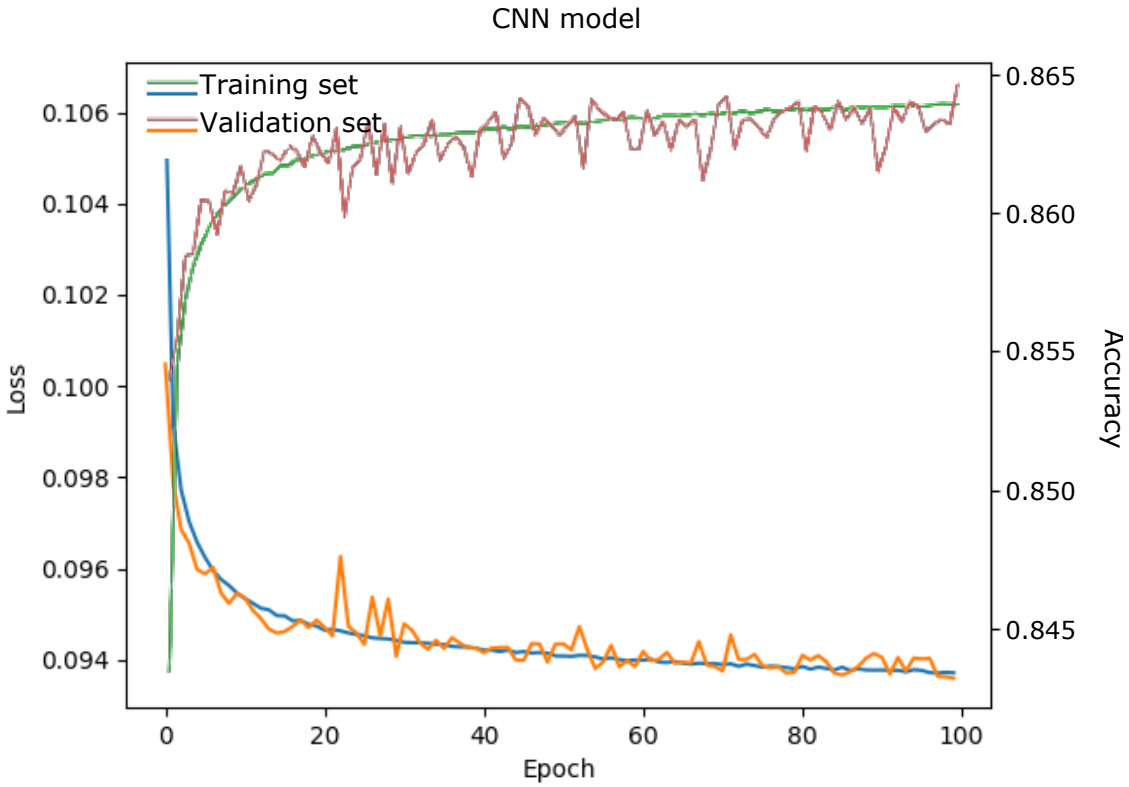


Figure 5.6 – *Decreasing loss (in X) and increasing accuracy (in Y) as a function of epoch for the training and validation set.*

optimizer [102]. A loss function is exploited to optimize those weights by minimizing the mean squared error between the predicted probability vector $\tilde{\mathbf{p}}$ and the corresponding ground truth vector $\mathbf{p}$ as follows :

$$\mathcal{L}_{cnn} = \frac{1}{L}\sum_{l=1}^{L} ||\mathbf{p}^l - \tilde{\mathbf{p}}^l||_2^2. \tag{5.3}$$

The batch size L is set to 128 instances (68×68 blocks) and the learning rate is equal to 10^{-3}. A hundred epochs are processed with a random shuffle of the dataset at each epoch. The CNN training was carried out on a RTX 2080 Ti GPU.

5.4 Experimental results

This section details the CNN performance through different aspects. First, the loss and accuracy are presented with the training and validation sets. Then, the CNN performance is analysed with the ROC curves through different splits and sizes. Finally, visual illustrations of partitions obtained from ground truth and CNN prediction are displayed.

5.4.1 Convolutional Neural Network performance

Figure 5.6 shows the loss and accuracy of the CNN model versus the training epochs on both training and validation sets. The blue and orange decreasing curves correspond to

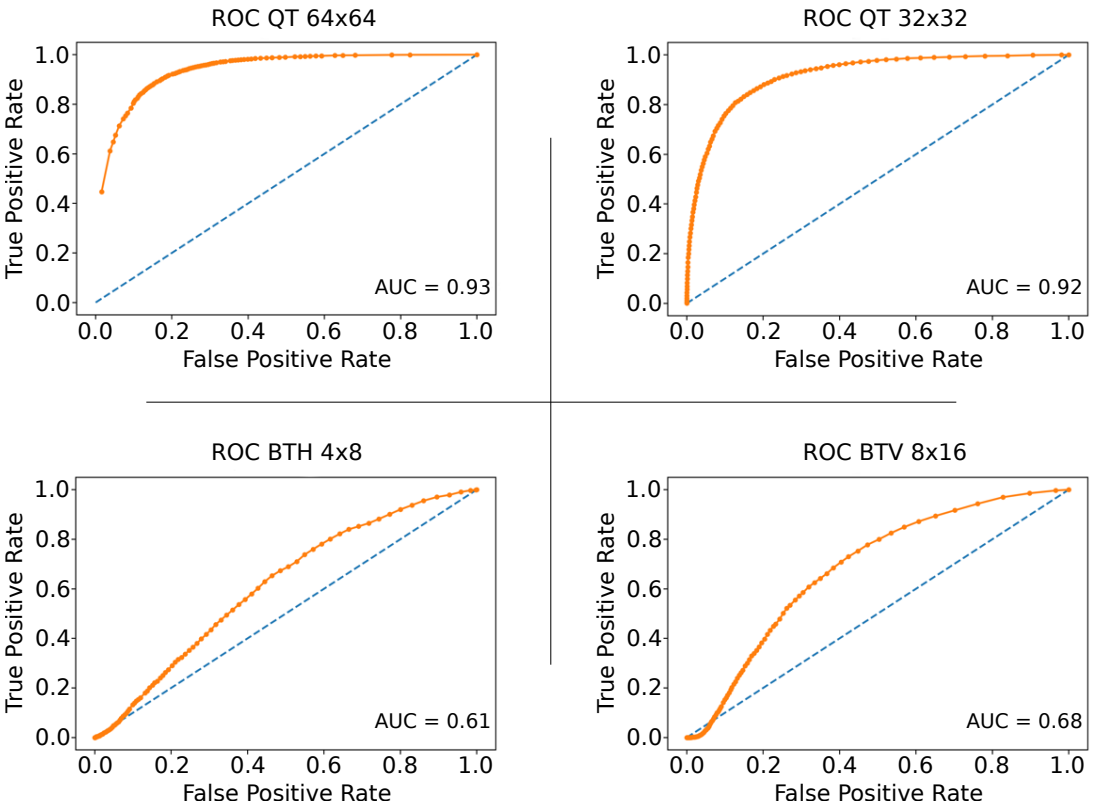


Figure 5.7 – *Several CNN ROCs for different splits and sizes on the CTC video sequences.*

the losses computed by Eq. (5.3) over hundred of epochs of the training. The green and red increasing curves represent the binary accuracies computed after a shrinkage with a threshold of 0.5. It means that if a predicted value is higher or lower than 0.5, the value is considered to be 1 or 0, respectively. This resulting value is compared with the ground truth and the accuracy is computed over all components of the vector of probabilities. The two curves in blue and green are from the training set and those in orange and red are from the validation set. The goal of the training is to update the model weights to minimize the loss at each iteration. We can notice that the loss curves decrease from 0.105 to 0.094 over 100 epochs for both training and validation sets. The validation curve follows the training one which means that the model generalizes well on the validation set. The model accuracy reaches more than 86% of good prediction, i.e., 86% of the estimated probabilities with a threshold of 0.5 are equal to the ground truth.

The CNN prediction performance is also analysed under the VTM with the ROC curves. The ROC curves represent the true positive rate versus the false positive rate. These curves select a target split to analyse versus all the other possible splits. The split probabilities required to plot these ROCs are determined by averaging all probabilities which are at the exact spatial position of the split. Figure 5.7 presents the ROC curves of QT, BTH and BTV splits at different CU sizes computed on the CTC sequences. The QT ROC curves show that the average probability is able to reach 0.8 of true positive for approximately 0.1 of false positive rate, for both 64×64 and 32×32 CU sizes. Instead, the BT ROC curves are closer to the random guess curve which is the diagonal line in blue. Indeed, the CNN pays more attention to the probabilities located on the first or second QT split. Moreover, the split choice for the high CU sizes is determined more easily as it concerns more pixels. The ROC area under curve values are given as a single score to compare the results among

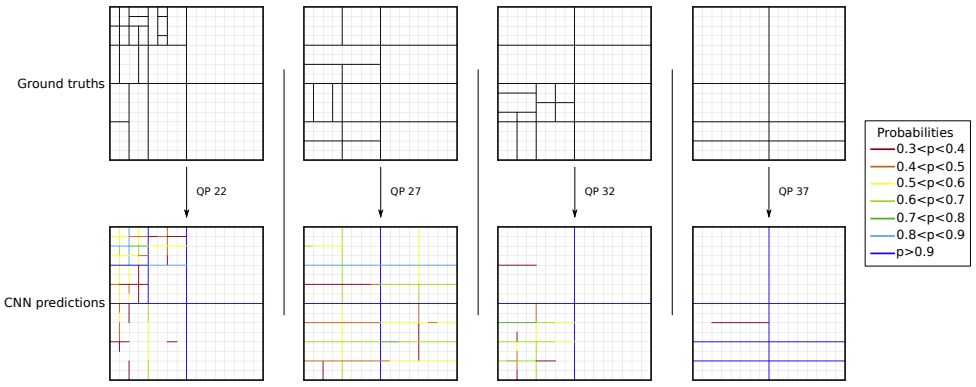


Figure 5.8 – *Instance of ground truth partitions and theirs corresponding CNN predictions for the sequence MarketPlace with QPs 22, 27, 32, and 37.*

the graphs. As shown by the curves, the ROC area under curve confirms the results by reaching scores higher than 0.9 for the QT and less than 0.7 for the BT splits.

The ground truths and predictions of the proposed CNN are illustrated on the first and second rows of Figure 5.8, respectively. On the top row, the optimal partitions derived by the VTM anchor are considered here as the ground truth. On the bottom row, the partitions predicted by the CNN are displayed with a colour code depending on their probabilities. The colour code varies from red to blue with a probability ranging between 0.3 and 1 and gray colour for probabilities below 0.3. The partitions for QP 37 is shallow with a maximum of three depths and two CUs of size 32×32. Its CNN predictions are relevant with each edge defined as a split with a probability higher than 0.9. For the other edges that are not defined as a split, the probabilities are lower than 0.3 except six edges predicted with probabilities between 0.3 and 0.4. Compared with the QP 37 partition, the QP 27 instance is partitioned deeper and is harder to predict efficiently. The figure shows that each edge determined as a split have a probability higher than 0.3 except the TT split. Furthermore, some CUs are predicted deeper.

5.5 Convolutional Neural Network inference optimisation

CNN brings great results for the complexity reduction of the VTM with a negligible BD-BR loss. However, CNN has known issue with inference complexity due to a high number of computations. Our CNN processes as input a luminance block of size 68×68 with 10 convolution layers and a final fully connected layer. Those layers combined with the others included in the CNN result in 226 088 training parameters. In this section, the execution time of the CNN inference is evaluated on CPU and GPU platforms.

5.5.1 Optimisation on Central Processing Unit

The framework Frugally-deep has been considered to generate the C++ source code for the CNN inference and to integrate it in the VTM. The proposed solution computed one CNN inference for each 64×64 block of pixels to predict their partitions. The execution time depends on the targeted CPU.

With frugally-deep, the CNN inference compiled without specific compilation options and with one thread lasts 153ms for a 64×64 block. The inference complexity is also studied under the tensorflow framework with different CPUs.

Table 5.2 – *Inference time under the tensorflow framework depending on the CPU exploited.*

CPU	Inference time
Xeon W-2125 (8 cores - 4 GHz)	2.13ms
Ryzen 5 2600X (6 cores - 3.6 GHz)	2.53ms
I7-8700 (6 cores - 3.2 GHz)	2.66ms
I5-10300H (4 cores - 2.5 GHz)	3.36ms

Table 5.3 – *Inference time under the frugally-deep framework depending on the activated compilation flag.*

Compilation flag	Inference time
-O0	113ms
-O1	4.06ms
-O2	3.72ms
-O3	3.68ms
-0fast	2.69ms

TABLE 5.2 lists the different CPUs used to infer with our CNN through the tensorflow framework and provides their respective inference times. The inference time depends mainly on the CPU clock rate. Indeed, with the Xeon W-2125 (8 cores - 4 GHz), the inference time to predict the partition of a 64×64 block is 2.13ms. The slowest CPU is the I5-10300H (4 cores - 2.5 GHz) with 3.36ms per inference. The inference complexity under the frugally-deep framework is at least $50\times$ more than the one under the tensorflow framework. It is mainly due to the performance optimisation of tensorflow added with the fact that the compilation does not processed optimisations with frugally-deep. Indeed, when activating several compilation flag it improves the performance in terms of complexity. Also some instruction extensions are available depending on the considered CPU.

TABLE 5.3 details the inference time of the CNN depending on the level of optimization specified by the compilation flags. In this study, the CPU used for the experiment is the Intel® CoreTM i5-10300H with quad core up to 2,5 GHz. The option *-march* is defined as native to exploit the instructions compatible with this CPU. The inference time to predict a 64×64 block is able to reach 2.69ms which is faster than when using tensorflow.

5.5.2 Optimisation on Graphics Processing Unit

The GPU is more adapted to train and infer CNN as most of the computation are matrix based and can be parallelized. Furthermore, the CUDA Application Programming Interface (API) that manages parallel computing plus the cuDNN framework that optimizes standard operation for CNN are available to improve the inference execution time. Different versions of tensorflow and CUDA are enabled which directly modifies the performance of the CNN. For these experiments the GPU selected is the Nvidia GTX 1650 Max Q. Under the tensorflow framework specialized for GPU version 2.0.0 and the CUDA version 10.0, the CNN infers at $254.65\mu s$ per 64×64 block partition which is equivalent to 7.7 FPS when a full HD resolution is considered.

The optimisation of the CNN model is proposed to improve the inference execution time. The TensorRT framework proposed by Nvidia defined different optimizations like layer or tensor fusion to optimize the GPU memory, bandwidth, and precision refinement by quantizing the CNN model.

Table 5.4 – *Inference time with TensorRT depending on the precision selected.*

TensorRT version	Inference time	FPS
TF-TRT 2.4.0	259.49μs	7.6 FPS
TF-TRT 2.0.0 (FP32)	135.17μs	14.5 FPS
TF-TRT 2.0.0 (FP16)	120.8μs	16.2 FPS

TABLE 5.4 details the inference time under the TensorRT framework with the GPU Nvidia GTX 1650 Max Q. Three configurations have been tested with different version of the TensorRT framework with 32-bit and 16-bit floating-point data types. The fastest configuration is the TF-TRT 2.0.0 with FP16 that is able to predict the partition of the 64×64 block partition at 120.8μs which is equivalent to 16.2 FPS on a full High Definition (HD) sequence. With FP32 configuration, the inference reaches 135.17μs.

These optimizations shown impressive results by using a GPU with TensorRT. Moreover, dedicated processor can be used for inference such as neural processing units proposed by Huawei or tensor processing units proposed by Google to reach even higher performance. Another option is to change the CNN architecture to limit the number of parameters and computations to reduce its inference execution time.

5.6 Conclusion

This chapter presented the basis of our complexity reduction methods that will be presented more thoroughly in the next chapters. Indeed, the CNN takes as input the current block and predicts all edges of each 4×4 block. Its performance influences the second part of our techniques and is critical to obtain interesting results. The prediction reaches 86% of good prediction with a threshold at 50%. Furthermore, the visual partition demonstrates that our CNN have relevant predictions.

The execution time of the CNN is negligible compared with the VTM encoding time. However, optimisation on CPU or GPU can be performed to limit the CNN inference complexity. The fastest configuration with the floating point set at 16 on the GPU Nvidia GTX 1650 Max Q reaches 16.2 FPS on a full HD sequence. Those inferences will be further exploited in the next chapters as information for the split decision.

CHAPITRE 6

Threshold based partitioning scheme for Versatile Video Coding intra encoders

6.1 Introduction

The previous chapter presented the basis of our complexity reduction solution. Indeed, a Convolutional Neural Network (CNN) is exploited in order to predict the partition of a 64×64 block. This chapter described the second part of our solution. This part processes the split decision to reduce the complexity while limiting the encoding quality loss. It takes as input the probabilities and outputs the split that will be considered in the Rate-Distortion Optimization (RDO) throughout the encoding.

In this chapter, we propose an efficient, threshold based complexity reduction technique for Versatile Video Coding (VVC) Test Model (VTM) intra encoders. The previously presented CNN is fed with a 68×68 pixels luminance block and it predicts a vector that gives probabilities of having edges at the 4×4 boundaries within this block. This probability vector is further compared with a threshold to skip unlikely splits. Two solutions are proposed to determine the threshold : uniform and non-uniform threshold selection. The first solution determines a uniform threshold for all the comparison. It reaches 42.2% complexity reduction for 0.75% Bjøntegaard Delta Bit Rate (BD-BR) loss. The second solution defines specific threshold for each split categories and block sizes relying on the BD-BR targeted which results in 54.5% complexity reduction for 1.33% BD-BR loss.

The remainder of this chapter is organized as follows. The proposed complexity reduction method is described in Section 6.2 and an analysis is introduced on the thresholds and their determinations to optimise their impacts on the complexity and the BD-BR. Section 6.3 presents the experimental setup and assesses the complexity reduction under the VVC Common Test Conditions (CTC). Finally, Section 6.4 concludes this chapter.

6.2 Partitioning prediction based on a Convolutional Neural Network

As presented in the previous chapter, our complexity reduction technique is relying on the CNN prediction. These predictions are exploited in this chapter as probabilities to define the split to test inside the encoder. Compared with a threshold, these split probabilities will give a decision in order to skip unlikely splits. Two propositions are detailed in this section

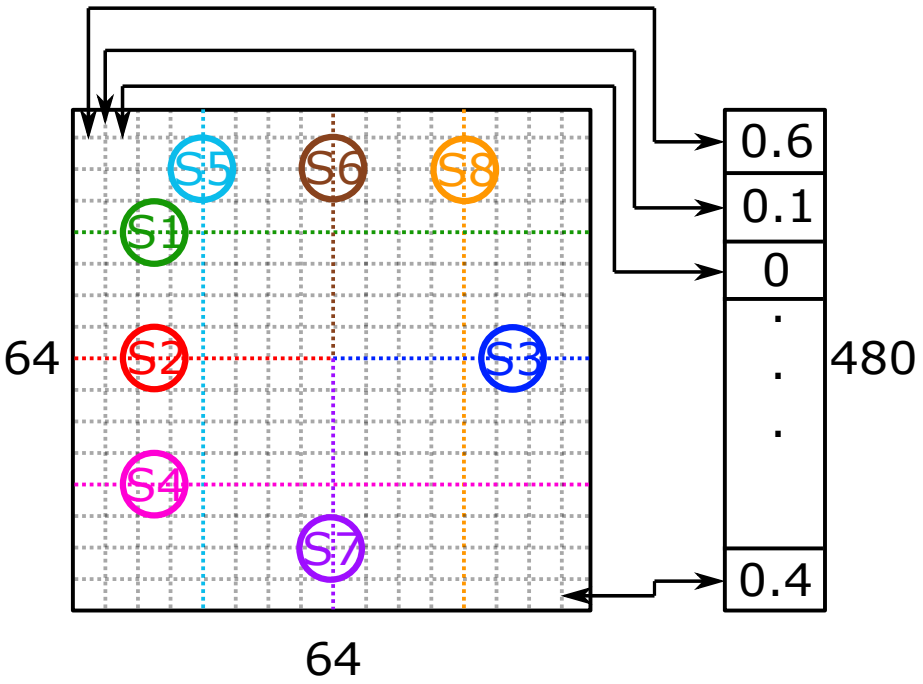


Figure 6.1 – *Correspondence between a predicted probability vector and a tree partitioning of a 64×64 block. Segments used to calculate the splits probabilities are also depicted in this figure.*

to define the threshold value with a uniform and non-uniform decision among the split and size. Furthermore, an analysis on the threshold is proposed to optimise the non-uniform threshold selection process.

6.2.1 Threshold-based decision

Figure 6.1 illustrates the segments exploited for the split decision process. The Quad-Tree (QT), Binary-Tree (BT), and Ternary-Tree (TT) probabilities are deduced as follows. The mean probabilities are first computed on the segments $\in [S1, S8]$, denoted by $P(S1)$ to $P(S8)$. To calculate the segment probabilities, for instance $P(S1)$, the boundary probabilities belonging to the segment position are averaged. The probability of the BTH split, denoted by $P(BTH)$, is the minimum between the probabilities of segments S2 and S3 as defined in

$$P(BTH) = min(P(S2), P(S3)). \tag{6.1}$$

The probability of the TTH split is computed with its respective segments as follows :

$$P(BTH) = min(P(S1), P(S4)). \tag{6.2}$$

The vertical splits, i.e., BTV and TTV splits, follow the same procedure but with vertical segments :

$$P(BTV) = min(P(S6), P(S7)), \tag{6.3}$$

and

$$P(TTV) = min(P(S5), P(S8)), \tag{6.4}$$

respectively.

Regarding the QT split probability, $P(QT)$ is calculated from the probabilities of the BTH and BTV splits as follows :

$$P(QT) = min(P(BTH), P(BTV)). \tag{6.5}$$

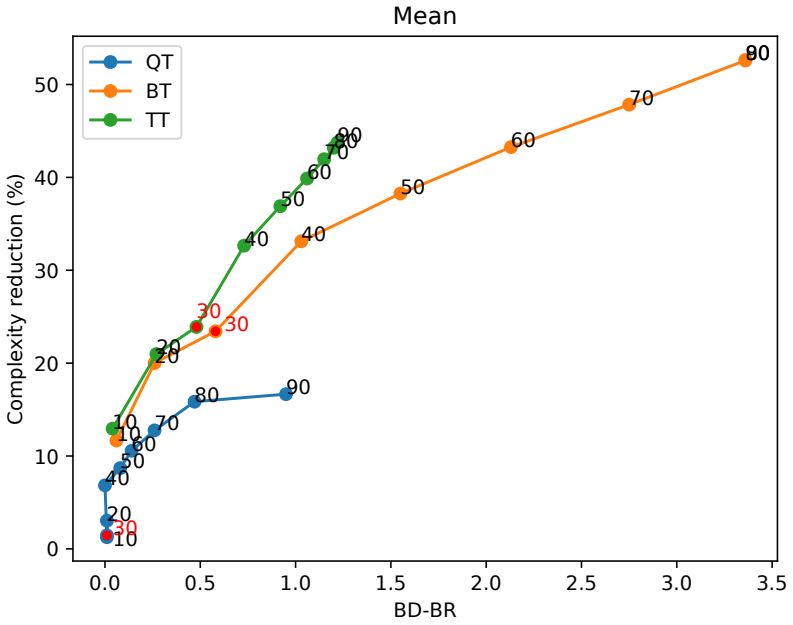


Figure 6.2 – *Average complexity reduction and BD-BR loss computed on the CTC when modifying β of each $P(S)$ independently.*

In all these cases, a split $S \in \{QT, BTH, BTV, TTH, TTV\}$ is skipped if $P(S)$ is below a predefined threshold value β. By means of this threshold, our solution is able to offer different rate-distortion-complexity trade-offs. The higher the threshold value β, the less splits are explored, which results in both higher complexity reduction and coding loss. Our solution supports a threshold value range of $\beta \in \{0\%, 100\%\}$.

6.2.2 Threshold analysis

This analyse is separated in two parts, a focus is performed firstly on the split and secondly on the Coding Unit (CU) size and split. To obtain the different rate-distortion-complexity trade-offs, the CTC sequences are encoded with the VTM. These encodings are performed with the application of the proposed complexity reduction method and several specific thresholds β to obtain the different trade-offs. Firstly, one threshold is specified for one split category, i.e., QT, BT, and TT. Secondly, one threshold is specified for each split category within each CU size. This second analysis gives more information as it is more specific, however the first one is interesting as it presents a global view of the splits impact.

Figure 6.2 presents the BD-BR loss in x-axis and the complexity reduction in y-axis when β of each $P(S)$ are modified independently. When the probability $P(S)$ is lower than β, the split S is skipped. $\beta \in [10, 90]$ are tested for the threshold as displayed in the figure for each split category which gives 9 points per split category. Three split categories are derived from the splits with QT, BT, and TT. Horizontal and vertical thresholds are combined for BT and TT. $\beta = 30$ which is the classical value presented in the uniform threshold method are displayed in red. The three curves have distinct shapes, the curve for the QT category has the lowest rate-distortion-complexity trade-off, while the curve for the BT category has the second one and the maximal trade-offs are obtained with the TT category. The BT has the maximal complexity reduction opportunity with more than 50%, however it comes at a high BD-BR loss of almost 3.5%.

In comparison, with $\beta = 90$ for TT, the BD-BR loss is lower, 1.22%, for a complexity reduction of 43.8%. With $\beta = 30$ both BT and TT have almost the same trade-offs with approximately 23% complexity reduction for 0.5% BD-BR loss. However, for QT, the probabilities are always higher than 40% which leads to negligible BD-BR loss and complexity reduction when $\beta = 30$. Having $\beta = 90$ for QT brings lower complexity reduction and higher BD-BR loss than setting $\beta = 20$ for BT or TT. These first analyses on the split categories prove that the trade-offs are different between those categories and that selecting distinct β for each category is more interesting than having a uniform β across the categories.

Figure 6.3 details the rate-distortion-complexity trade-offs relying on the classes introduced in the CTC. In majority the three curves of each class have approximately the same shape. QT has the lowest trade-offs followed by BT and TT. However, depending on the sequences, relying on their resolutions or characteristics, the trade-offs are similar across the split categories. For instance, the class A2 and E have the curves shape that are almost the same which means that the impact of the splits is similar. Nevertheless, a difference is noted on the proportion of the impact, skipping the QT with $\beta = 90$ is approximately the same as skipping BT or TT with $\beta = 20$. Skipping BT reduces more the complexity than skipping TT but for a highest BD-BR loss.

A detailed analysis on the split is further conducted. It evaluates the split category with QT, BT, and TT at each CU size. Indeed, the CU size when the QT is available as a split is between 64×64 to 16×16, for BT the size is between 32×32 to 8×8, and for TT the size allowed is between 32×32 and 16×16. For rectangular block, to facilitate the analysis, the values are categorised as a squared block with the maximum of its height or width, i.e., if a block is 16×8, the block is categorised as 16×16.

Figure 6.4 presents the complexity reduction and BD-BR loss for the QT through the several block sizes defined previously. The analysis on the split categories shown that QT has the lowest trade-offs. However, the results across the CU sizes show that the curves have different shapes. The size 32×32 is the only size with QT that has interesting trade-offs. Indeed 10% complexity reduction can be achieved with less than 0.1% BD-BR loss. A maximum of 18% complexity reduction is reached for a BD-BR loss of 0.5% with $\beta = 90$. When the size is fixed at 64×64 the BD-BR loss is important for the low complexity reduction opportunity, the misleading of the QT split at size 64×64 has an important impact on the overall loss. In fact, as the QT is the only split available at this size, if the QT is skipped, then, no other split can be performed to limit the BD-BR loss so the block is stopped at the size 64×64. Finally, for the size 16×16, a small proportion is selected as QT. Indeed, the other splits are available at this size and QT is not always available which leads almost no complexity reduction and BD-BR loss.

Figure 6.5 focuses on the BT split category with horizontal and vertical BTs under the three predefined sizes. The shape of the two curves that represent the sizes 32×32 and 16×16 are similar. At their maximum, these curves reach more than 30% complexity reduction for less than 1.3% BD-BR loss independently. Moreover, approximately 15% complexity are reduced for 0.2% BD-BR loss when $\beta = 30$ for both curves. However, for the size 8×8, the complexity reduction is lower than 7% for all thresholds. This size is not often represented as the blocks included in this category are 8×8, 8×4, and 4×8 plus the complexity of those sizes are lower than high block sizes. Another remark is that the predicted probabilities of the BT split at those sizes are often lower than 30 which makes the complexity reduction constant after this threshold.

Figure 6.6 studies the TT split which has the most opportunity when comparing the splits impact. When $\beta \leqslant 30$, the two curves have almost the same shape with a maximum

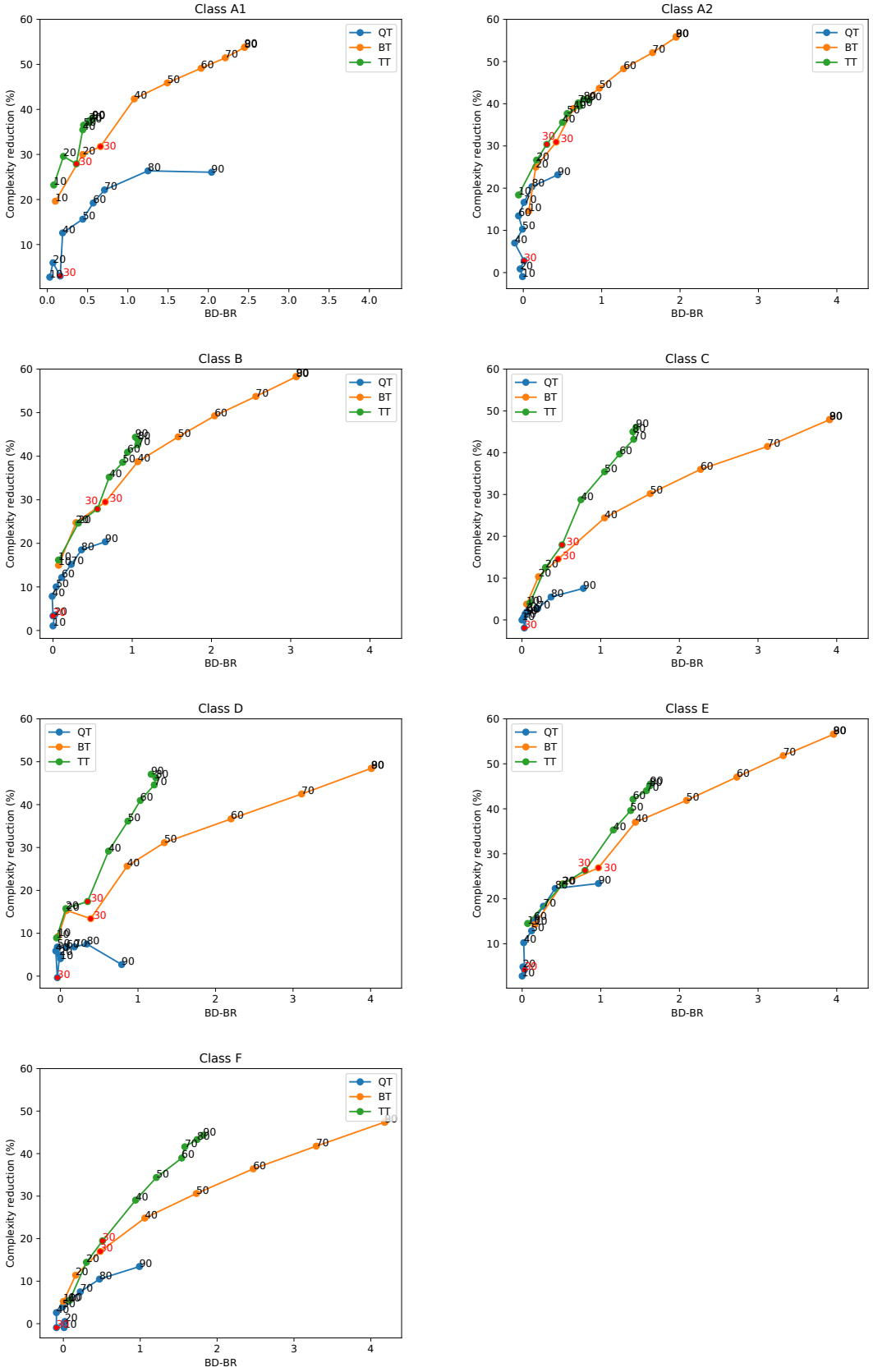


FIGURE 6.3 – *Complexity reduction and BD-BR loss of each CTC class when modifying β of each $P(S)$ independently.*

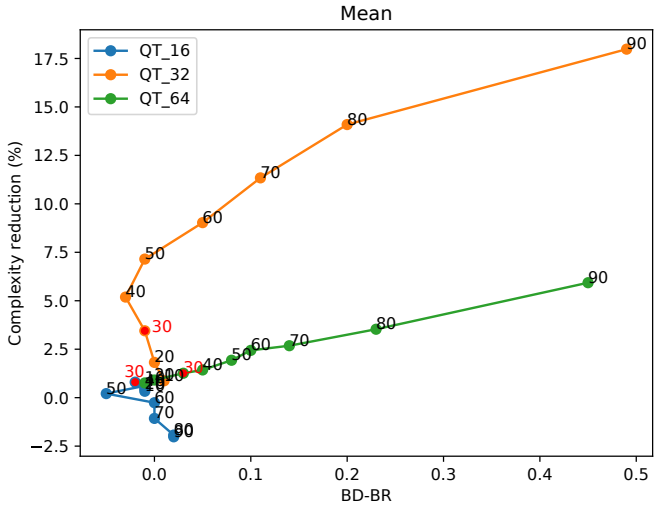


Figure 6.4 – *Average complexity reduction and BD-BR loss computed on the CTC when modifying the threshold of $P(QT)$ for each size independently.*

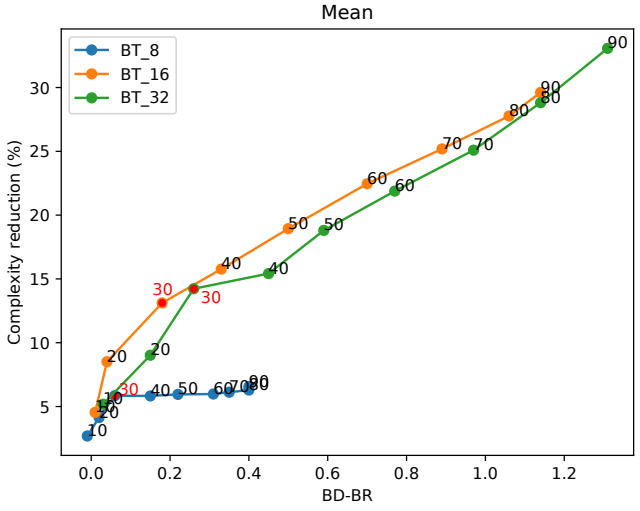


Figure 6.5 – *Average complexity reduction and BD-BR loss computed on the CTC when modifying the threshold of $P(BT)$ for each size independently.*

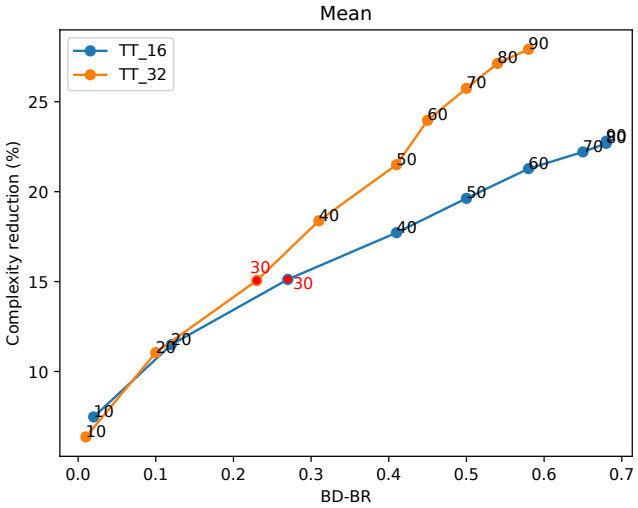


Figure 6.6 – *Average complexity reduction and BD-BR loss computed on the CTC when modifying the threshold of $P(TT)$ for each size independently.*

of 15% complexity reduction for 0.25% BD-BR loss. However, when $\beta > 30$, the curves diverged and the TT at size 32×32 has better trade-offs than the ones with the size 16×16. When $\beta = 90$ for size 32×32, the complexity reduction reaches 28% for less than 0.6% BD-BR loss.

To conclude those analyses, the size 32×32 for each split category brings the most complexity reduction opportunity but at a high BD-BR loss. This can be explained by the representation of 32×32 block under the VTM plus the complexity to process this block size. The size 64×64 or 8×8 have almost no impact on the complexity reduction but lead to a significant BD-BR loss. For size 64×64 the probabilities predicted by our CNN are always high which limits the QT skip. However, when QT is skipped, the impact on BD-BR is important due to the impossibility to pursue the tree partitioning process. As shown before with the analysis of the split, BT has the highest complexity reduction opportunity and the highest BD-BR loss followed by TT and QT even when the size are distinct. Nevertheless, the best trade-off is obtained by TT at size 32×32 with $\beta = 90$ which results in 28% complexity reduction for 0.58% BD-BR loss. Those analyses demonstrate the impact of each split and the impact of each split for a defined size. However, the impacts are defined only separately. Indeed, the complexity reduction and BD-BR loss can not be simply added between two configurations as each configuration has impact on others, i.e., if all the splits are skipped at size 32×32 the tree partitioning is stopped and no check will be further performed whereas if one split is allowed the split selection can be sub-optimal but at least the tree partitioning process will continue on lower sizes.

6.2.3 Threshold selection

Relying on the above analysis, the selection of distinct threshold for each split is crucial to optimize the split choice. Indeed, the CNN predicts probabilities that are dependent of the position inside the 64×64 computed block. Some parts of our prediction may have lower accuracy than others due to the difficulty of predicting particular region of the block. Generally, the probabilities that are on small blocks are more challenging, for instance predicting the QT on 64×64 block is easier than the BT on 8×4 block. This is related on

the impact of the number of probability throughout the CNN training. One probability for the block 8×4 has less impact on the training than 32 probabilities for the 64×64 block. Relying on those facts, an algorithm is further designed to select the thresholds linked to their repercussions on the BD-BR and the complexity reduction.

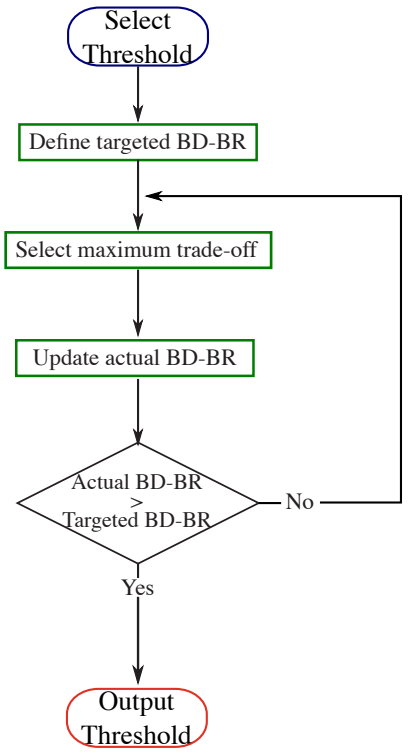


Figure 6.7 – *Local search algorithm based on gradient descent to define the threshold β.*

Figure 6.7 presents the block diagram of our method to select optimal threshold based on the previous analysis. Our solution optimizes the thresholds β relying on their trade-offs which is computed through the ratio Δ_{CQ} defined as follows :

$$\Delta_{CQ} = \frac{\Delta T}{\Delta BD - BR}, \tag{6.6}$$

where ΔT is the difference of complexity reduction between the considered solution and the one tested with the previous β value. $\Delta BD - BR$ is the difference of bitrate degradation between the considered solution and the one tested with the previous β value.

First, the user determines a targeted BD-BR which characterizes the maximum BD-BR loss that will be achieved. Then, the ratio Δ_{CQ} is computed for all the thresholds for each split and size. It selects the maximum trade-off and updates the temporary BD-BR named as actual BD-BR. Once this actual BD-BR is superior to the targeted BD-BR the algorithm is terminated and the different β are outputted. If not, the previous process is repeated and the trade-offs are updated based on the one selected. Once the thresholds β are defined for each split at each size, they are included into the VTM for comparison with the split probability $P(S)$. As the thresholds β are selected based on the average study across the CTC classes, the usage of the CTC to optimize our thresholds is negligible. Indeed, the thresholds are not specified for each class or content which make them generic.

Table 6.1 – *Performance of the proposed solution in VTM6.1 with different uniform threshold values β in comparison with state of the art techniques.*

Class	Lei et al. [104], VTM3.0		Park et al. [105], VTM4.0		Proposed $\beta = 10$		Proposed $\beta = 20$		Proposed $\beta = 30$	
	BD-BR	ΔET	BD-BR	ΔET	BD-BR	ΔET	BD-BR	ΔET	BD-BR	ΔET
Class A1	0.79%	44.9%	0.67%	32.0%	0.35%	45.3%	0.87%	56.3%	1.55%	62.9%
Class A2	0.96%	39.5%	1.07%	33.0%	0.30%	40.1%	0.83%	52.6%	1.47%	60.0%
Class B	1.06%	45.1%	0.98%	33.0%	0.26%	36.9%	0.75%	51.5%	1.41%	61.1%
Class C	1.09%	48.3%	1.17%	35.0%	0.15%	13.9%	0.56%	26.4%	1.20%	37.9%
Class D	0.97%	44.2%	0.88%	35.0%	0.08%	13.0%	0.33%	22.7%	0.83%	32.5%
Class E	1.32%	47.9%	1.34%	34.0%	0.42%	29.5%	1.18%	43.8%	2.29%	54.4%
Mean	1.03%	45.0%	1.02%	33.7%	0.26%	29.8%	**0.75%**	**42.2%**	1.45%	51.5%
Class F	Nan	Nan	Nan	Nan	0.21%	15.2%	0.75%	25.4%	1.61%	36.3%

6.3 Experimental results

This section details the experimental setup and analyses our results over the state of the art techniques. Several configurations of our solution are assessed in order to propose different rate-distortion-complexity trade-offs. Furthermore, two techniques are proposed independently. Our first solution calculates split probabilities based on the CNN prediction and skips the split if the probability is inferior to the threshold β which is defined uniformly. The second solution performs the same process by exploiting the CNN prediction but the thresholds are defined non-uniformly related to the analyses previously presented.

6.3.1 Experimental setup

All our experiments were conducted under All Intra (AI) configuration with VTM version 6.1. Each encoding and CNN prediction were carried out individually on Intel Xeon E5-2603 v4 processor running at 1.70 GHz on Ubuntu 16.04.5 operating system. The CNN inference is carried out in C++ through the frugally-deep library [103] with a trained Python model. The CTC with the four Quantization Parameter (QP) values (22, 27, 32, 37) are used to compare our results with the state of the art techniques. The complexity reduction is calculated through ΔET determined in Eq. 2.7 and the coding quality is measured with the BD-BR.

The execution time of the CNN is not included in T_C as it highly depends on the processor performance and can be neglected. Indeed, a complexity analysis and optimisation of the CNN inference has been presented on the previous chapter and the inference complexity of the CNN reaches 16.2 Frames per Second (FPS) for a full High Definition (HD) sequence.

6.3.2 Complexity reduction performance under the Versatile Video Coding test model

There are several existing techniques that seek to reduce the complexity of the Multi-Type Tree (MTT) partitioning search in the VTM. The most competitive ones have been implemented by Lei et al. [104] in VTM3.0 and Park et al. [105] in VTM4.0. Thereafter, various new coding tools have been adopted to VTM, such as low frequency non-separable transforms [23], intra sub-partitioning [106], and joint chroma residual coding [107]. However, the tree partitioning search is kept unchanged, so the comparison between our technique in VTM6.1 and the techniques proposed in [104], [105] remains still relevant and fair.

TABLE 6.1 presents our results with uniform thresholds such as $\beta = 10, 20, 30$ over the techniques described in [104], [105]. The first proposed configuration with $\beta = 10$ limits

Table 6.2 – *Description of the thresholds for the configuration C1 and C2 under the several splits and sizes available.*

Configuration	QT			BT			TT	
	64 × 64	32 × 32	16 × 16	32 × 32	16 × 16	8 × 8	32 × 32	16 × 16
C1	10	70	30	30	30	30	80	20
C2	10	70	30	30	20	30	60	20

the BD-BR loss to only 0.26% with 29.8% complexity reduction. The second configuration with $\beta = 20$ still features a negligible BD-BR loss of 0.75% and a high complexity reduction of 42.2%. Our last configuration with $\beta = 30$ halves the complexity for 1.45% BD-BR loss.

In VTM3.0, Lei et al. [104] achieved approximately the same complexity reduction but for a loss of 0.28% BD-BR compared with our second configuration. Park et al. [105] obtained 33.7% complexity reduction for 1.02% BD-BR loss, so it lags behind our second solution ($\beta = 20$) both in terms of complexity and BD-BR.

Our solution achieves better coding performance on high resolution sequences especially on classes A and B. For instance, our proposal with $\beta = 10$ almost halved the complexity with only 0.35% BD-BR loss for class A1. For $\beta = 30$, the complexity reduction reaches 62.9% with 1.55% BD-BR loss. For low-resolution C and D classes, our solution is able to reduce the complexity by 35.2% with 1.01% BD-BR loss. The training database is mainly composed of 4K and Full HD images, so our CNN performs better on these resolutions.

The previous solution presented was based on uniform value of β that manages to obtain different rate-distortion-complexity trade-offs which are not optimal. Indeed, the threshold is shared across the splits and the CU sizes whereas it has different impacts on the BD-BR or the complexity reduction.

Figure 6.8 plots the relative complexity reduction (ΔET) versus BD-BR loss over the CTC classes A-E (marked with crosses) for our proposed four uniform configurations ($\beta = 10, 20, 30, 40$), our two non-uniform configurations, and the state of the art techniques [104], [105]. Tackling the VVC coding complexity is of particular importance to higher resolutions so the corresponding results are also separately given for classes A-B (marked with circles). The two non-uniform configurations are $C1$ and $C2$ which correspond to a targeted BD-BR of 1.2 and another targeted BD-BR of 1. The thresholds for both configurations $C1$ and $C2$ are detailed in the TABLE 6.2 through the different splits and sizes. $C1$ reached 54.5% complexity reduction for 1.33% BD-BR loss whereas $C2$ brought 50.1% complexity reduction for 1.16% BD-BR loss. Those two configurations with non-uniform β across split and size provide higher complexity reductions and lower BD-BR losses compared with our solution with uniform β. For instance $C1$ has 3% more complexity reduction and 0.13% lower BD-BR loss than the uniform configuration with $\beta = 30$. As presented before, solution with uniform threshold already surpasses the state of the art methods, so, our non-uniform solution also outperforms those methods. The comparison between the two non-uniform solutions is interesting as we can notice that β specified for QT has not been modified and the TT with a block with one of its sides at 32 is the more adjusted. As previously presented, QT is the last split that the algorithm will modified as it does not bring the highest trade-off. Instead, TT has the most interesting trade-offs which enables the algorithm to adjust its thresholds more easily.

With high-resolution sequences, the complexity savings of our uniform and non-uniform proposals range from 40.8% to 66.7% with a BD-BR loss from 0.30% to 2.37%, respectively. With the uniform configuration $\beta = 20$, the overall complexity reduction is 42.2% for only 0.75% BD-BR loss. For high resolution classes A-B, the respective metrics are 53.5% and

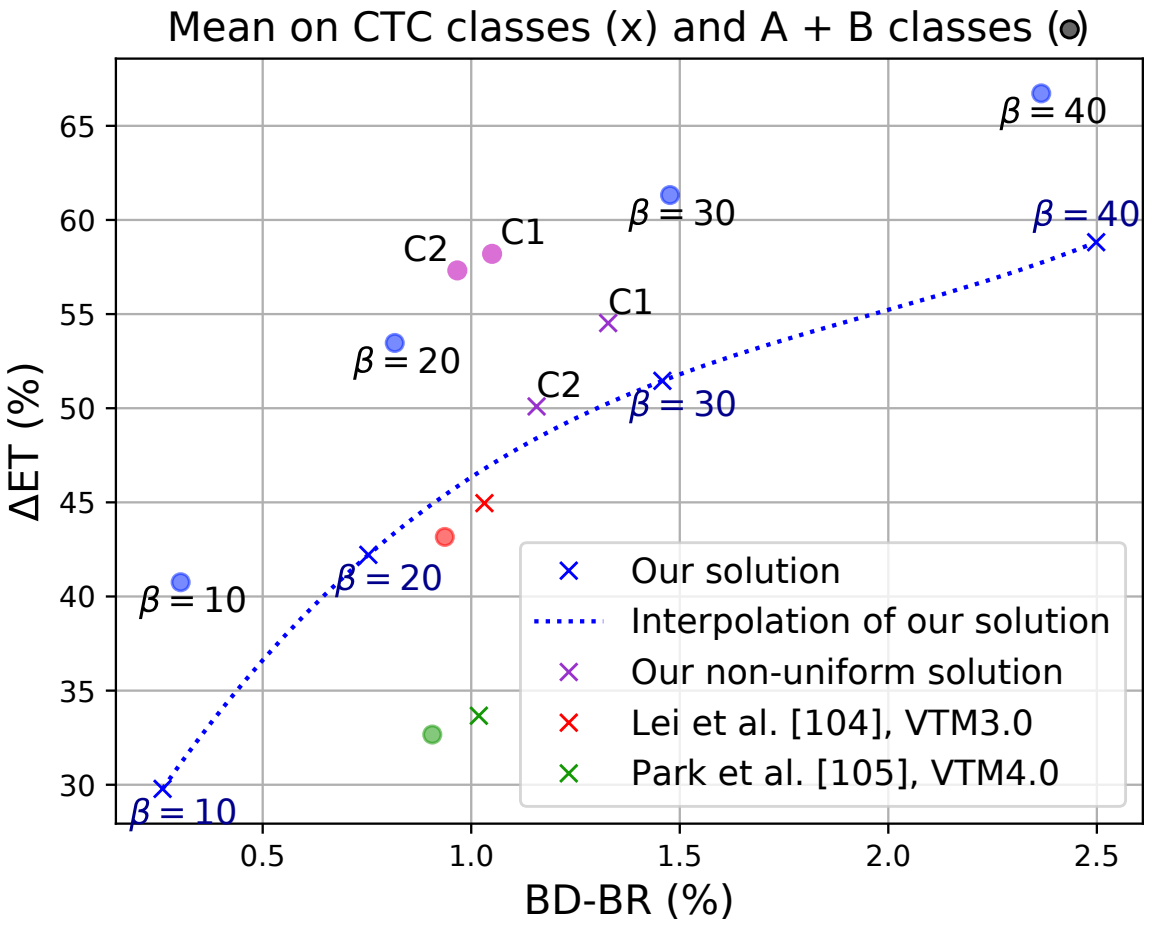


Figure 6.8 – *Performance comparison between our proposed uniform solution, non-uniform solution, and state of the art techniques. Circles are averaged results of Full Hd and 4K sequences. Crosses are averaged results of CTC classes without class F.*

0.82%. Compared with the previous configuration ($\beta = 20$), [104] achieves 10.3% lower complexity reduction for 0.12% more BD-BR loss. Our solution also outperforms [105] in terms of both BD-BR and complexity reduction. Our non-uniform solutions obtain also higher complexity reductions and lower BD-BR losses on the high resolution classes than on the CTC average. The trade-offs of our non-uniform $C1$ and $C2$ solutions are also better than our uniform configurations and the state of the art methods.

To conclude, our uniform and non-uniform configurations are able to achieve higher complexity reduction and BD-BR gain than [104], [105] with high-resolution sequences, comparable results with smaller resolutions, and averagely better coding performance with the entire CTC test set. Multiple β values also make our implementation more configurable to various operating points, like practical video encoders with several presets.

6.4 Conclusion

This chapter presented a CNN-based complexity reduction technique for the VVC reference encoder VTM6.1 under AI configuration. The CNN is used to analyse the texture inside each 64×64 coding block and to predict vector probabilities for 4×4 boundaries inside these blocks. From the probability of boundaries, a split probability is deduced and compared with a uniform threshold β. A non-uniform threshold determination is also proposed that makes the threshold evolves through the split categories and the CU sizes. In VTM6.1 intra coding, the proposed uniform solution enables 42.2% complexity reduction for a slight BD-BR loss of 0.75%. With high-resolution sequences, the speed-up is even higher, up to 54.5% complexity reduction at the cost of 0.85% BD-BR loss. The non-uniform configuration reaches 54.5% complexity reduction for 1.33% BD-BR loss. Our proposal allows several configurations and the majority of them overcome the state of the art techniques.

CHAPITRE 7

Machine learning based Quad-Tree-Multi-Type Tree partitioning scheme for Versatile Video Coding intra encoders

7.1 Introduction

Chapter 5 has presented a Convolutional Neural Network (CNN) which is exploited as a basis for our complexity reduction solution detailed in Chapter 6. This CNN takes as input a block to predict its partition through a vector of probabilities. The average probability over edges of the split is then compared with a threshold to decide whether to perform the split or not at each depth. The main drawback of this solution is its local decision which does not leverage all probabilities of the block edges. In this chapter, all the block edge probabilities are taken into account in order to improve the accuracy of the splits decision.

This work proposed an efficient complexity reduction technique for the Quad-Tree (QT)-Multi-Type Tree (MTT) partitioning. Our approach combines a moderate complexity CNN that extracts spatial features of a block of pixels followed by a Multi-Class Classifier (MCC) that derives the more likely splits to be evaluated by the Rate-Distortion Optimization (RDO) process. A single CNN is used to process a 64×64 block and outputs the probability of each boundary of all the 4×4 pixel blocks within this input. At each level of the hierarchical tree partitioning process, a MCC is used to predict from the set of boundary probabilities, the N-most likely splits to explore by the encoder. This part takes advantage of all the CNN probabilities within the current block to predict the split probabilities. One MCC is trained for each size of the 16 different block sizes. At each depth, the number of tested splits N can be adjusted from one to the total number of possible splits. This allows exploring a wide range of trade-off between the complexity reduction and the quality loss. The proposed solution with the top-$N = 3$ configuration reaches 46.6% complexity reduction for a negligible Bjøntegaard Delta Bit Rate (BD-BR) loss of 0.86%. The top-$N = 2$ configuration enables on average a complexity reduction of 69.8% for 2.57% BD-BR loss.

The rest of this chapter is organized as follows. Section 7.2 describes the proposed two-stage method based on the combination of two machine learning algorithms. Section 7.3 presents the performance of the Decision Tree (DT) models and a comparison of our method against state of the art techniques in terms of both complexity reduction and BD-BR loss. Finally, Section 7.4 concludes this chapter.

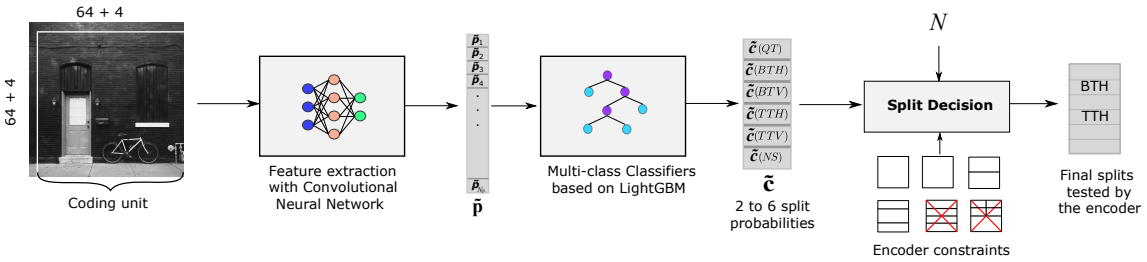


FIGURE 7.1 – *Workflow diagram of the proposed tree partitioning scheme for VVC All Intra (AI) coding. The input luminance block is first processed by a CNN to predict* $\tilde{\mathbf{p}}$*, a vector of* N_p *probabilities describing all edges at each* 4×4 *sub-block. The vector* $\tilde{\mathbf{p}}$ *is then processed by a decision tree LightGBM to predict probabilities of the six partitioning options through the vector* $\tilde{\mathbf{c}}$*. The top-N splits with the highest probabilities are tested by the Rate-Distortion (RD) process of the encoder to select the optimal split in terms of RD-cost.*

7.2 Partitioning prediction based on Machine Learning

As described previously, Versatile Video Coding (VVC) introduces Binary-Tree (BT) and Ternary-Tree (TT) splits at the cost of an high increase in computational complexity. This recursive tree partitioning process computes the rate-distortion cost for a set of coding tools at each Coding Unit (CU). The proposed tree partitioning prediction technique is composed of a CNN for spatial features extraction followed by a set of MCCs based on DTs to predict the appropriate split decision at different depths of the partitioning tree.

7.2.1 Overall presentation

Figure 7.1 illustrates the flow diagram of the proposed tree partitioning prediction solution. The CNN presented previously in the Chapter 5 is exploited in order to give partition information to the split decision block based on MCC. Such as the previous contribution, the CNN takes as input the current 68×68 block and outputs a probabilities vector. The MCCs are then fed with the probabilities vector $\tilde{\mathbf{p}}$ derived from the CNN to predict the split decision that will be performed at each tree depth. The predicted vector $\tilde{\mathbf{c}}$ is computed as follows :

$$\tilde{\mathbf{c}} = g_{\omega_i}(\tilde{\mathbf{p}}), \quad \forall i \in \{1, \ldots, M\}, \tag{7.1}$$

where g_{ω_i} is a parametric function of the MCC i, ω_i is its training parameters, and M is the number of considered MCCs.

A separate MCC based on DT model is applied for each CU size. The MCC takes as input the probability vector $\tilde{\mathbf{p}}$ and predicts a vector $\tilde{\mathbf{c}}$ of six probabilities that correspond to the six possible splits performed by the VVC encoder at each tree depth. This approach results in M separate MCC models that are trained separately to enhance the prediction performance and enable better convergence of the model with reduced number of training parameters. Once the split probabilities are derived for the CU, a selection of the highest probabilities is made based on the selected configuration. The configuration specifies the N best values which define the number of tested splits by the encoder. Thus, the encoder performs only the N most likely splits corresponding to the highest probabilities (Top-N) predicted by the MCC. The remaining splits with lower probabilities than the top-N candidates are skipped by the encoder to reduce the encoding computational complexity.

7.2.2 Multi-class classifiers based on Decision Tree models

The decision approach adopted in the previous Chapter 6 only considers the probability at the spatial location of a specific split. For instance, the BTH split takes into account only the probabilities of the horizontal edges in the middle of the CU. By leveraging only a restricted location of the CU, an important information is omitted to determine the final split decision. In order to take advantage of all probabilities inside the current CU, the DT model is adopted to process the vector of probabilities predicted by the CNN model. This vector of probabilities can be considered as spatial features relevant for the tree partitioning process. Therefore, the CNN inference is carried-out once for each block of size 64×64 to predict the corresponding probability vector $\tilde{\mathbf{p}}$, then a specific DT model processes this vector at each level of the partitioning tree to derive a set of N more likely splits to explore by the encoder. To predict split probabilities at various CU sizes, multiple models are considered covering all possible sizes of the rectangular sub-blocks from 64×64 to 4×4 excluded. As illustrated in Table 7.1, sixteen CU sizes can be further split in six to two different partitions including the no split option. The DT model is fed with the probability vector $\tilde{\mathbf{p}}$ and the Quantization Parameter (QP) value. The probability vector is then cropped into a sub-vector that includes only the probabilities of the sub-block edges. The DT model performs a multi-class classification by predicting a probability vector $\tilde{\mathbf{c}}$ of six components corresponding to the probabilities of the six possible splits. Therefore, the probabilities of non possible splits are set to zero during the training process.

Several machine learning models have been investigated to solve this multi-class classification problem including DT, random forest, Support Vector Machine (SVM) with different kernels, and LightGBM (LBGM) model [108]. This latter has been retained for its good classification performance associated with its low complexity at both training and inference stages.

Table 7.1 – *Possible splits according to the CU size in addition to the no-split option*

Height \ Width	64	32	16	8	4
64	QT	-			
32	-	All	BT, TT	BT	BTH
16		BT, TT	All	TTH	TTH
8		BT, TTV		BT	BTH
4		BTV, TTV		BTV	-

7.2.3 Training process for Decision Tree LightGBM model

The DT models are implemented under the LBGM framework [108] version 2.3.1. This latter is a gradient boosting framework based on decision tree developed by Microsoft. LBGM has many advantages such as its low memory usage, the capacity of handling large-scale data, and the low inference computational time. This last advantage is essential for our problem as the inference is carried-out at each CU level.

LBGM is a DT method that sums the predictions of all the trees to get a richer interpretation of the problem. The trees are optimized in a stage-wise way by adding or updating one tree sequentially based on the error of the whole ensemble learned so far.

For DT classification, the used cross-entropy loss function is defined as follows

$$\mathcal{L}_{dt} = -\frac{1}{L}\sum_{l=1}^{L} \mathbf{c}^l \cdot \log \tilde{\mathbf{c}}^l, \tag{7.2}$$

where $\mathbf{c}$ is the vector of ground truth split probabilities and $\tilde{\mathbf{c}}$ is the vector of split probabilities predicted by the model. The symbol $\cdot$ stands for the dot product.

7.3 Experimental results

In this section, the experimental setup and the performance of the proposed method are assessed. The performance of the MCCs is presented through their top-N accuracies. The proposed complexity reduction method is then assessed in terms of both computational complexity reduction and BD-BR loss with respect to the VVC Test Model (VTM). Multiple configurations of our method are explored depending on the tested top-N DT LBGM output splits. There are several existing techniques that have investigated the complexity reduction induced by the new MTT partitioning in VVC. The proposed solution is compared with four best performing state of the art techniques including solutions proposed by Fu *et al.* [56], Chen *et al.* [57], Yang *et al.* [58], and Li *et al.* [61]. Finally, the inference overhead of both CNN and DT models are assessed.

7.3.1 Experimental setup

All experiments are conducted with the VVC Test Model (VTM) version 10.2 in AI coding configuration. We consider test video sequences defined in the VVC Common Test Conditions (CTC) [19]. The Common Test Conditions (CTC) video sequences are separated in seven classes as follows : A1 (3840×2160), A2 (3840×2160), B (1920×1080), C (832×480), D (416 × 240), E (1280 × 720), and F (832 × 480 to 1920 × 1080). These video sequences are encoded at four QP values : 22, 27, 32, and 37.

The proposed solution is assessed in terms of both coding efficiency and computational complexity. The coding efficiency is measured with the BD-BR metric [10] that computes the bitrate loss over four QPs in percentage with respect to the anchor (ie. VTM10.2) for the same Peak Signal-to-Noise Ratio (PSNR) objective quality. The BD-BR is calculated across the three components Y, U, and V. The computational complexity reduction with respect to the anchor is assessed by computing the ΔET as described in Eq. 2.7.

Our complexity reduction technique has been integrated in the VTM10.2 encoder which is developed in C++ programming language. The CNN is built and trained with Python under the Keras framework and then the model is converted to C++ code with the frugally deep framework [103]. The DT models are also trained in python and then converted to C++ with the LBGM framework [108].

All encoding operations were carried out sequentially on an Intel Xeon E5-2603 v4 processor running at 1.70 GHz under Ubuntu 16.04.5 operating system (OS).

7.3.2 Decision Tree LightGBM performance

The second part of the proposed solution includes DT LBGM models that take advantage of all spatial features predicted by the CNN to derive split probabilities. Multiple models are trained in order to handle the different CU sizes. The input element range starts from one probability plus the QP for 4×8 or 8×4 block sizes to 480 probabilities plus the QP for 64×64 CU size. The output is a vector of six classes with QT, the two BTs, the two

Table 7.2 – *Performance of every DT LBGM models with their defined size and number of output through top-1, top-2, and top-3 accuracy on the validation set.*

Width	Height	Number of classes	Top-1 accuracy (%)	Top-2 accuracy (%)	Top-3 accuracy (%)
64	64	2	91.69	-	-
32	32	6	59.94	78.38	88.87
32	16	5	58.50	77.96	89.85
16	32	5	56.55	77.48	89.24
32	8	4	54.80	80.22	94.11
8	32	4	54.91	80.45	94.40
32	4	3	66.64	87.80	-
4	32	3	68.80	87.38	-
16	16	6	50.95	71.27	84.60
16	8	4	62.89	82.74	94.05
8	16	4	62.36	83.25	94.39
16	4	3	68.96	90.12	-
4	16	3	68.95	88.54	-
8	8	3	81.46	93.05	-
8	4	2	82.16	-	-
4	8	2	86.26	-	-
Average		2	86.70	-	-
		3	70.96	89.38	-
		4	58.74	81.67	94.24
		5	57.53	77.72	89.55
		6	55.45	74.83	86.74
		-	67.24	82.97	91.19

TTs and no split. For CUs with less splits available due to VTM restrictions, a mask is applied on the predicted vector to restrict the unsuited splits.

TABLE 7.2 presents the accuracy of the DT LBGM models on the CTC validation set. Three values are reported to analyse the DT LBGM results based on the top-N accuracy with $N \in \{1, 2, 3\}$. Top-N accuracy is a metric that measures how often the correct class falls in the top-N highest predicted probabilities. Top-1 accuracy reaches on average 67.24% of correct prediction. Based on the number of available splits the prediction is different. It decreases on average from 86.7% to 55.45% for 2 classes and 6 classes, respectively. The highest top-1 accuracy is achieved with the 64×64 block size binary classification between QT and no split with 91.69%. All DT LBGM models have more than 50% top-1 accuracy even with multi-class classification. The top-2 accuracy is presented for models with at least three classes, which reaches on average 82.97%. The lowest accuracy is 71.27% obtained with the lowest block size available for six classes decision, i.e., 16×16. Finally, top-3 accuracy achieves 91.19% accuracy on average over the model with at least four classes. By skipping half of the available splits, the top-3 configuration achieves 86.74% accuracy for 6 classes. This enables to reduce the complexity by a factor of two with a high confidence on predicting the correct split. Theses results exhibit that the accuracy of DT models with 3 classes depends on the available splits. In fact, higher accuracies are reached for the 8×8 block size model in top-1 and top-2 by at least +12% and +3%, respectively, compared with other models with 3 classes. In addition to the no split mode, two splits are in competition : either BT and TT in the same direction or BT horizontal and vertical. As shown by the results, the direction of a split is easier to predict than the difference between BT and TT in the same direction.

Table 7.3 – *ΔET and BD-BR performance of the proposed solution in comparison with state of the art techniques in AI coding configuration.*

Class	Sequence	Fu *et al.* [56], VTM1.0		Chen *et al.* [57], VTM4.0		Li *et al.* [61], VTM7.0		Ours top-3, VTM10.2		Ours top-2, VTM10.2	
		BD-BR	ΔET	BD-BR	ΔET	BD-BR	ΔET	BD-BR	ΔET	BD-BR	ΔET
Class A1	Tango2	1.47%	54%	1.38%	48.7%	1.52%	40.8%	0.54%	46.7%	1.51%	66.4%
	FoodMarket4	1.01%	53%	0.84%	30.3%	1.26%	42.2%	0.53%	47.5%	1.45%	66.2%
	Campfire	1.06%	46%	1.29%	49.1%	2.02%	48.7%	0.74%	48.3%	2.01%	67%
	Average	1.18%	51%	1.17%	42.7%	1.6%	43.9%	0.6%	47.5%	1.66%	66.5%
Class A2	CatRobot1	1.58%	43%	1.99%	50.4%	2.16%	45.4%	0.99%	46.1%	2.6%	66.9%
	DaylightRoad2	1.49%	49%	2%	54.1%	1.16%	49.4%	1.02%	50.6%	2.55%	73.5%
	ParkRunning3	0.58%	51%	0.8%	40.6%	1.15%	41.7%	0.38%	42.6%	0.95%	59.5%
	Average	1.22%	47.7%	1.6%	48.4%	1.49%	45.5%	0.8%	46.4%	2.03%	66.6%
Class B	MarketPlace	0.60%	52%	-	-	0.8%	46.6%	0.53%	55.6%	1.36%	75.8%
	RitualDance	1.08%	45%	-	-	1.07%	44.9%	0.87%	51.7%	2.5%	73%
	Cactus	0.92%	44%	1.73%	49.8%	1.12%	49.3%	0.89%	49.4%	2.65%	72.4%
	BasketballDrive	0.89%	49%	1.54%	50.1%	1.64%	52%	0.92%	51.6%	2.57%	73%
	BQTerrace	0.78%	48%	1.4%	56%	1.11%	45.6%	1.06%	46.5%	2.82%	72%
	Average	0.85%	47.6%	1.56%	52%	1.15%	47.7%	0.85%	51%	2.38%	73.2%
Class C	RaceHorses	0.77%	42%	1.35%	50.6%	0.96%	46.5%	0.66%	46.2%	2.08%	74.5%
	BQMall	0.94%	39%	2.12%	58.9%	1.17%	49.8%	1.01%	47%	3.2%	75.8%
	PartyScene	0.53%	42%	1.01%	51%	0.61%	45.2%	0.58%	44.1%	2.21%	74.3%
	BasketballDrill	1.91%	46%	2.05%	54.8%	1.63%	39.3%	1.6%	45.4%	4.75%	73%
	Average	1.04%	42.3%	1.63%	53.8%	1.09%	45.2%	0.96%	45.7%	3.06%	74.4%
Class D	RaceHorses	0.91%	39%	1.28%	54.7%	1.2%	41.6%	0.67%	43.6%	2.46%	65.6%
	BQSquare	0.51%	44%	0.75%	52.8%	0.74%	44.5%	0.61%	44.4%	2.49%	69.7%
	BlowingBubbles	0.61%	37%	1.4%	54.9%	0.92%	41.6%	0.64%	40.5%	2.48%	64.9%
	BasketballPass	0.75%	43%	1.77%	53.1%	1.41%	44.5%	0.9%	44.5%	2.94%	65.9%
	Average	0.7%	40.8%	1.3%	53.9%	1.07%	43.1%	0.7%	43.3%	2.59%	66.5%
Class E	FourPeople	1.26%	41%	2.71%	56.3%	1.33%	49.9%	1.2%	44.7%	3.64%	71%
	Johnny	1.44%	39%	2.77%	55.8%	2.33%	48.2%	1.42%	42.6%	3.97%	67.4%
	KristenAndSara	1.27%	40%	2.17%	52.8%	1.76%	50.5%	1.06%	44.4%	3.47%	68.5%
	Average	1.32%	40%	2.55%	55%	1.81%	49.5%	1.23%	43.9%	3.69%	69%
Average		1.02%	44.8%	1.62%	51.2%	1.32%	45.8%	0.86%	46.6%	2.57%	69.8%
Class F	ArenaOfValor	-	-	-	-	-	-	1.08%	22%	3.38%	49.7%
	BasketballDrillText	-	-	2.09%	56.3%	-	-	1.59%	22.5%	4.69%	52.5%
	SlideEditing	-	-	0.52%	45.4%	-	-	1.21%	27.3%	4.67%	58.8%
	SlideShow	-	-	2.11%	45.8%	-	-	1.5%	26%	4.85%	56.2%
	Average	-	-	1.57%	49.2%	-	-	1.34%	24.5%	4.4%	54.3%

7.3.3 Complexity reduction performance under the Versatile Video Coding test model

The two-stage proposed model is integrated in the VTM10.2 encoder configured in AI setting. For fair comparison, the reference used to compute the BD-BR loss and the complexity reduction is the classical VTM encoder including multiple native speed-up techniques [8] for the tree partitioning process. These speed-up techniques reduce significantly the execution time with a slight BD-BR degradation compared with an exhaustive RDO process. Thus, these experiments exhibit the gain provided by our approach compared with the common configuration of the VTM encoder.

To compare our method, the best performing state of the art techniques in terms of complexity reduction and BD-BR loss are selected. These state of the art techniques are not performed under the same VTM version, however, the tree partitioning tool has not been significantly changed during the standardization.

TABLE 7.3 presents the BD-BR and complexity reduction performance of our method compared with the state of the art techniques proposed by Fu *et al.* [56], Chen *et al.* [57] and Li *et al.* [61]. In this table, two configurations are presented based on the top-2 and top-3 configurations. The results are illustrated for the CTC sequences from class A1 to class F. The average is presented for each class independently and for all the sequences from class A1 to class E. Class F is not taken into account in the average as it includes specific sequences such as screen content.

On average, through all the sequences our top-3 configuration reaches 0.86% BD-BR loss for a complexity reduction of 46.6%. Compared with Fu *et al.* [56] and Li *et al.* [61] methods, our solution achieves better performance in terms of both BD-BR and complexity

reduction. Chen *et al.* [57] solution reduces 4.6% more complexity but at the expense of a significant increase in loss of 0.76% in BD-BR compared with our top-3 configuration. This low gain in the complexity reduction is not relevant compared with the loss in BD-BR which is almost doubled. The second configuration with top-2 achieves on average 2.57% BD-BR loss with 69.8% of complexity reduction.

High video resolutions are the main interest in the VVC development. In fact, tackling high resolution complexity is particularly important as the encoding time is proportional to the sequence resolution. Both A and B classes represent high resolution sequences with 4K and full HD, respectively. Our top-3 and top-2 configurations are able to reach 47% and 66.6% complexity reductions for 0.7% and 1.85% BD-BR losses on class A sequences, respectively. In both BD-BR and complexity reduction metrics our method is better than Chen *et al.* [57] and Li *et al.* [61] techniques. Compared with these two techniques, our top-3 configuration solution achieves on average a lower loss by 0.69% and 0.85% in BD-BR and a higher computational complexity reduction by 1.5% and 2.3%, respectively. Fu *et al.* [56] solution enhances the complexity reduction by 2.4% at the expense of a high BD-BR loss of 0.5% compared with our top-3 configuration. Concerning the class B sequences, our top-3 configuration is able to halve the complexity reduction with 51% of complexity reduction for less than one percent of BD-BR loss, i.e., 0.85%. The closest to our method is Fu *et al.* [57] technique achieving similar BD-BR score but with lower complexity reduction by 3.4%. Our top-2 configuration enables 73.2% complexity reduction for 2.38% BD-BR loss. The highest performance in terms of both complexity reduction and BD-BR loss is obtained by the *MarketPlace* sequence. Indeed, the top-3 and top-2 configurations reach 55.6% and 75.8% complexity reductions for 0.53% and 1.36% BD-BR losses, respectively.

For lower resolution classes C to E, the top-3 configuration achieves on average less than 1% BD-BR loss for a maximum of 47% complexity reduction. Compared with Fu *et al.* [56], our method has lower BD-BR loss for higher complexity reduction through all low resolution classes. Li *et al.* [61] has higher BD-BR loss for lower complexity reduction for class C and D compared with our top-3 configuration. For class E, Li *et al.* [61] solution achieved 5.6% more complexity reduction but at the cost of 0.58% BD-BR loss compared with our method. Chen *et al.* [57] almost doubled the BD-BR loss with 1.76% for a complexity reduction increase of 9.9% compared with our top-3 configuration through low resolution classes. In the case of the top-2 configuration, the performance is lower on low resolutions than on high resolutions. In fact, this approach on low resolution contents including classes C, D, and E reaches the complexity of high resolution contents (classes A1, A2, and B) of 70.1% for a higher BD-BR loss of 3.06% which is higher by approximately 1% compared with the BD-BR loss of high resolutions contents (2.09%). Low resolution contents result in deeper partitions, so the more the complexity is reduced, the less space is available to skip splits. As the global partition is composed of more splits, the impact on BD-BR is more significant at high complexity reductions.

The class F has specific sequences with screen contents such as slides or gaming contents. Our method is not optimized for this type of contents, since these contents have not been considered in the learning process. However, our approach still achieves 24.5% complexity reduction for 1.34% BD-BR loss. The work of Chen *et al.* [57] is the only one that reported results on class F with three out of four sequences. On average, they got higher complexity reduction but for a higher BD-BR loss. Adding screen content to the training dataset could be a solution to enhance the trade-off between computational complexity reduction and coding efficiency loss.

Figure 7.2 shows the performance of our method in complexity reduction versus BD-BR jointly with state of the art techniques in a 2D plan averaged on the CTC classes

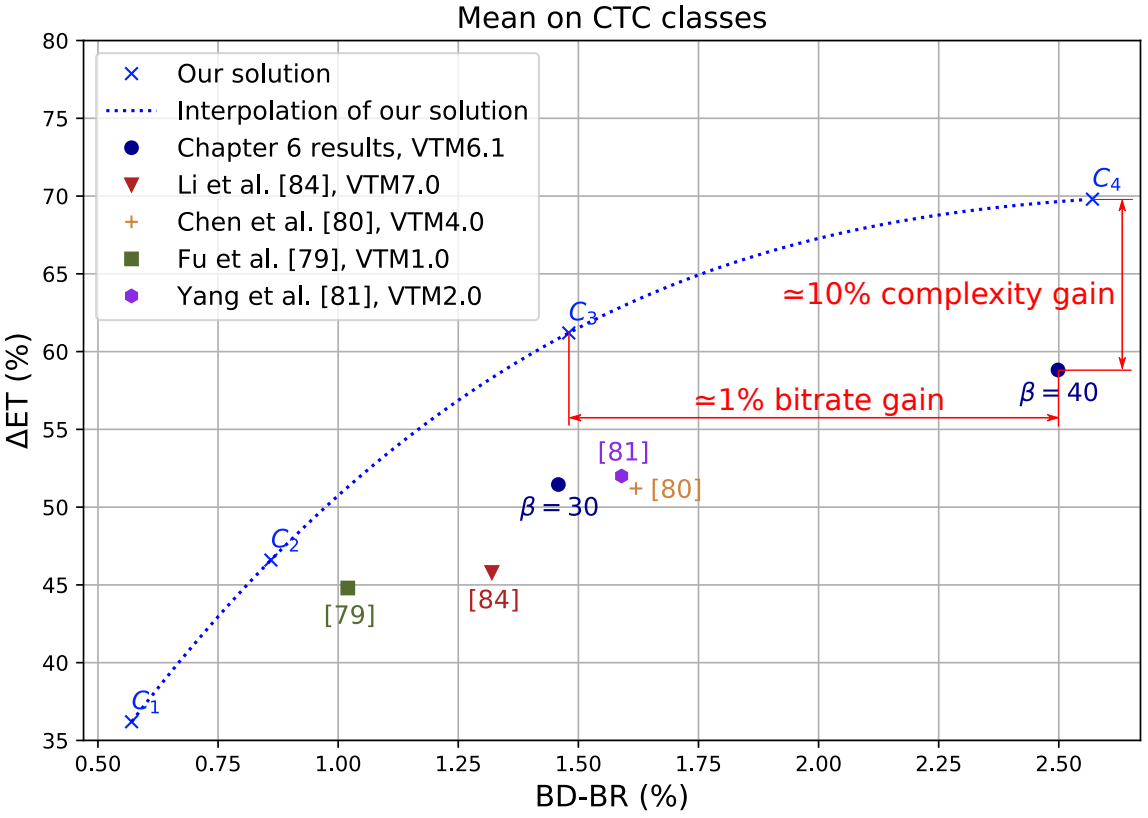


Figure 7.2 – *Complexity reduction versus BD-BR performance comparison between the proposed method and the state of the art techniques averaged on the CTC classes without class F (AI configuration). An interpolation curve over our four configurations is plotted in blue dot line.*

Table 7.4 – *Overheads of the CNN and DT LBGM (in %) for C_2 configuration with respect to the execution time of the VTM anchor.*

	Class A1	Class A2	Class B	Class C	Class D	Class E	Class F	Average
CNN overhead	2.7%	1.2%	1%	0.6%	0.5%	1.4%	1.2%	1.2%
DT overhead	2%	1.7%	1.7%	1.6%	1.6%	1.8%	1.6%	1.7%
Total	4.7%	2.9%	2.7%	2.2%	2.1%	3.2%	2.8%	2.9%

without class F. In addition to the configurations presented in the table, two new specific configurations have been included. The selection of top-3 for the multi-classification with 6 outputs (classes) and top-4 for the other DT LBGM models is defined as C_1. The top-3 configuration is defined as C_2. The top-3 for all DT LBGM models except the multi-classification with 6 outputs for which top-2 is used in configuration C_3, and finally, top-2 for all DT models is defined as C_4. These several configurations reach different trade-off points between complexity reduction and BD-BR loss, which allows us to draw an interpolation curve over these four configurations. This interpolation helps us to compare the results since the rate-distortion-complexity trade-off points are not linear. It allows a comparison at each point of complexity reduction or BD-BR loss of our method. As explained through the previous table, the figure confirms that our solution outperforms the best performing state-of-the-art techniques. The points representing the state of the art solutions are all below the interpolation curve of our approach. At the same BD-BR loss, we achieve a complexity reduction of 6.5%, 12.5%, 10.1%, and 12.1% compared with Fu *et al.* [56], Li *et al.* [61], Yang *et al.* [58], and Chen *et al.* [57] techniques, respectively. For the same complexity reduction, our solution enables a BD-BR gain of 0.21%, 0.49%, 0.54%, and 0.61% over Fu *et al.* [56], Li *et al.* [61], Yang *et al.* [58], and Chen *et al.* [57] techniques, respectively.

The solution proposed in the Chapter 6 is also illustrated in this figure to show the benefit of our new proposal in terms of complexity and coding performance. Two configurations of the previous solution were selected with decision thresholds : $\beta = 30$ and $\beta = 40$. Our C_3 and C_4 configurations are better than the $\beta = 30$ and $\beta = 40$ solutions with a gain in complexity reduction of approximately 10% for a similar BD-BR loss. Compared with $\beta = 40$ which reaches 58.8% complexity reduction, our solution C_3 affords a significant BD-BR gain of 1.02%.

7.3.4 Complexity overhead

In this section, the complexity overhead of the CNN and DT inferences is investigated under the VTM. TABLE 7.4 presents the time spent in the CNN and in the DT predictions for the C_2 configuration in comparison with the VTM10.2 anchor encoding time. These values are obtained by applying a ratio between the CNN or DT time and the VTM reference encoding time. The execution time of the CNN inference is on average lower than the execution time of the DT model. Indeed, even if the CNN inference is more complex, the DT infers at each CU size unlike the CNN which is carried-out only once for each 64×64 CU. The execution time ratio of the CNN is higher in the Ultra High Definition (UHD) classes especially the class A1. This can be caused by the early skip methods integrated in the VTM that lead to shallow partition, i.e., less time in the encoding process. Moreover, the results for the DT inference time is homogeneous through all the CTC classes. On average, both execution times of the CNN and DT are under 2% and taken together, they require less than 3% of the encoding time. These results show that the prediction times are negligible, especially since the CNN and DT models can be optimized, accelerated, or run

in parallel with the encoding. Indeed, the execution time of the CNN can be significantly reduced by targeting a Graphics Processing Unit (GPU) or on multi-core processors with optimizations such as presented in Chapter 5.

7.4 Conclusion

In this chapter we have proposed a two stage CNN and DT method to reduce the complexity of the VTM encoder in AI configuration. The CNN is designed to predict a CU partition through a vector of probabilities based on the local activity of pixels in a block. This vector considered as spatial features is then passed as input to the DT model that predicts the split probabilities at each CU depth. The DT method integrated after the CNN inference benefits from all the probabilities inside the computed CU instead of taking only the probabilities at the spatial location of the split. Depending on the selected configuration, a top-N probability selection is performed on the DT output to skip the unlikely splits.

Our proposed method outperforms state of the art techniques in terms of trade-off between complexity reduction and coding loss. Concerning the top-3 configuration, our proposal assessed on the VVC CTC sequences enabled on average 46.6% complexity reduction for a negligible BD-BR loss of 0.86%. The top-2 configuration was able to reach a higher complexity reduction of 69.8% for 2.57% BD-BR loss. These promising results motivated us to extend our method to the Random Access (RA) configuration. This extension investigated as future work will require to take advantage of the motion flow among adjacent frames.

CHAPITRE 8

Machine learning based Quad-Tree-Multi-Type Tree partitioning for Versatile Video Coding inter coding

8.1 Introduction

Chapter 7 has presented a solution based on Convolutional Neural Network (CNN) and Decision Tree (DT) techniques to reduce the complexity of the Versatile Video Coding (VVC) Test Model (VTM) under All Intra (AI) configuration. This chapter focuses on the encoding complexity reduction under the Random Access (RA) configuration through the tree partitioning process pruning. The tree partitioning selection is different under the RA configuration compared to the AI configuration due to the inter prediction tools added in the encoding process. Indeed, the RA configuration takes advantage of the temporal relationship between the frames in addition to the the spatial correlation. Inter coding has specific tools that benefit from previously encoded frames through inter prediction modes. Several tools are available under inter coding for the prediction, however, in this solution only the classical Motion Vector (MV) is exploited. The MV defines the movement between a reference frame and the frame currently encoded. This movement is described with the Picture Order Count (POC) of the reference frame and the two values characterizing block motion flow in the x and y-axis. Several MVs are computed to select the optimal one through different reference frames and movements. Predicting the partition under RA configuration is a challenging task since it relies on more information than under the AI configuration.

In this chapter, an efficient complexity reduction method is proposed relying on Machine Learning (ML) algorithms. First, a CNN predicts the partition based on its input, a $128 \times 128 \times 3$ luminance pixel values. The input is the current Coding Tree Unit (CTU) processed plus the two reference CTUs obtained by the inter prediction mode process. These two reference CTUs provide the inter prediction information that are essential to deduce the partition. The outputted vector represents the partition of the processed CTU. Then, to optimise the information provided by the CNN and gives more accurate decision, DTs are exploited at each depth level. Indeed, the tree partitioning information computed by the CNN is given as input to the DTs depending on their location and size. Several DTs are available to predict the probabilities of each split such as a multi-classification task. At each Coding Unit (CU) size, a DT is trained as the inputs are different from one another. Indeed, DT models are able to predict the split probabilities from 128×128 to 4×8 or 8×4 CU sizes. Finally, the number of split tested or skipped depends on the targeted

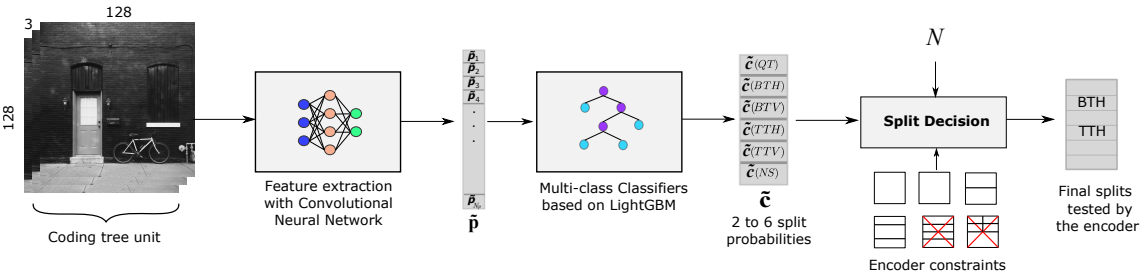


Figure 8.1 – *Workflow diagram of the proposed fast tree partitioning scheme for VVC under RA configuration. The current luminance CTU B_{cur} plus its two CTUs of reference B_{MV1} and B_{MV2} which are processed by a CNN to predict $\tilde{\mathbf{p}}$, a vector of N_p probabilities describing all edges at each 4×4 block. The vector $\tilde{\mathbf{p}}$ is then processed by the MCC to predict probabilities of the six partitioning options through the vector $\tilde{\mathbf{c}}$. The top-N splits with the highest probabilities are tested by the Rate-Distortion (RD) optimization of the encoder to select the optimal split in terms of RD-cost.*

complexity reduction-Bjøntegaard Delta Bit Rate (BD-BR) trade off. The splits with the highest probabilities are tested by the VTM encoder to minimize the BD-BR loss and maximize the complexity reduction.

The remainder of this chapter is organized as follow. The proposed method is described in Section 8.2 with its ML algorithms. Section 8.3 details the dataset used to train the CNN and the DT models. The training parameters of the CNN and DT models are presented in Section 8.4. The performance evaluation of our solution in terms of both complexity reduction and BD-BR loss is proposed in Section 8.5 alongside with the CNN and DT accuracy results. Finally, Section 8.6 concludes this chapter.

8.2 Proposed method

To reduce the complexity of the VTM under inter coding, a method based on ML technique is proposed to skip unlikely splits. This section describes this ML-based method relying on a CNN that computes features related to the tree partitioning and then the Multi-Class Classifier (MCC) predicts the split probabilities using those previous features.

8.2.1 Overall presentation

The proposed method reduces the complexity by skipping unlikely splits inside the VTM encoder process. The characterization of a split as unlikely is predicted with ML techniques. Figure 8.1 details the global scheme of the proposed method to reduce the complexity of the VTM under the RA configuration. The ML techniques are separated in two parts. Firstly, a CNN predicts a vector of features that represents the CTU tree partitioning. Secondly, the MCC determines a probability for each split available based on the features of the processed block. Finally, relying on predefined configurations, the VTM performs the top-N best splits to maximize the complexity reduction while minimizing the BD-BR loss.

Description of the inputs

These ML techniques are taking advantage of inter prediction and the pixels of the current block. Let B_{cur}, the considered CTU, a block of 128×128 pixels. B_{MV1} and B_{MV1}

are defined to describe the two CTUs of reference that correspond to the two MVs with the lowest RD-costs.

A pre-process is necessary to obtain the three components B_{cur}, B_{MV1}, and B_{MV2} of the CNN input. Before the tree partitioning process, the inter prediction method computes different blocks from several encoded frames to find the most related block and estimates the RD-cost of the current CTU. Indeed, the two MVs with the lowest RD-costs are used to find the two most related block of size 128×128, B_{MV1} and B_{MV2}. This is the classical process of the VTM, no complexity overhead is added. The pixels of those two blocks are selected to bring the information of the inter prediction. Finally, the luminance pixels, which results from a block $\mathbf{B} = \{B_{cur}, B_{MV1}, B_{MV2}\}$ of dimension $128 \times 128 \times 3$, are given to the CNN as input in order to extract features related to the partition of the current CTU.

Feature extraction with Convolutional Neural Network

To extract the features required to predict the unlikely splits, a CNN proceed the 3-component input $\mathbf{B} = \{B_{cur}, B_{MV1}, B_{MV2}\}$ in order to take advantage of the inter prediction. The CNN outputs the N_p-length vector $\tilde{p} = \left[\tilde{p}_0, \ldots, \tilde{p}_i, \ldots, \tilde{p}_{N_p-1}\right]$ where each element $\tilde{p}_i$ represents the probability associated with the 4×4 edge i of the CTU. From Eq. 5.2 (page 77), the vector length N_p, is equal to 1984 for a CTU size of 128×128. The features extraction block processes the 3-components input $\mathbf{B}$ to predict the vector of probabilities $\tilde{\mathbf{p}}$

$$\tilde{\mathbf{p}} = f_\theta(\mathbf{B}). \tag{8.1}$$

where f_θ is a parametric function of the CNN with training parameters θ.

Split probability determination with Multi-Class Classifier

The MCC is fed with the probability vector $\tilde{\mathbf{p}}$ derived from the CNN to predict the split decision performed at each depth based on the following expression :

$$\tilde{\mathbf{c}} = g_{\omega_i}(\tilde{\mathbf{p}}), \quad \forall i \in \{1, \ldots, M\}, \tag{8.2}$$

where g_{ω_i} is a parametric function of a classifier i, ω_i are its training parameters, and M is the number of considered classifiers.

One classifier is available for each block size. Indeed, for a 128×128 block decision, the MCC takes as input the whole vector with the 1984 features. However, with a 8×4 block, the MCC takes as input only the edge inside this block which results in 1 feature. Based on the considered features, the MCC predicts the split probabilities. These split probabilities are gathered inside the vector $\tilde{\mathbf{c}}$. The splits supported in $\tilde{\mathbf{c}}$ are Quad-Tree (QT), Binary-Tree (BT)H, BTV, Ternary-Tree (TT)H, TTV, and no split.

Split decision

Finally, the VTM encoder selects the top-N best split that will be computed. However, the encoder has constraints on the available splits which depend of the CU size and the previously computed split. For instance, after a BT or a TT split, QT is not available. Based on these constraints, a selection of the top-N splits are then proposed, depending on the selected configuration. For instance, if the top-2 configuration is selected, the two highest probabilities from $\tilde{\mathbf{c}}$ which are available relying on the VTM constraints are tested by the encoder. As the unlikely splits are not performed, the complexity is reduced with the aim to limit the BD-BR loss. A more specific description of the CNN and the ML models are proposed thereafter.

8.2.2 Convolutional Neural Network

The CNN exploited in this method is using the architecture of MobileNetV2 [109]. This architecture is designed for mobiles with limited resources. Indeed, this model decreases the number of operations and the memory resources which are key aspects for our problem. The architecture starts with a fully convolutional layer of 32 filters followed by 19 residual bottleneck layers. The non-linearity is computed via ReLU6, the kernel size is 3×3 plus dropout, and the batch normalization is performed during training.

A modification of the original architecture is proposed in our method to predict the 1984 features related to the tree partitioning. The input is $\mathbf{B}$, the $128 \times 128 \times 3$ block of pixels. To determine the vector $\tilde{\mathbf{p}}$ that contains the features, a fully connected layer is computed as the last layer. Moreover, the Quantization Parameter (QP) which has an important impact on the tree partitioning is given as an external input to the fully connected layer. Indeed, having a low QP influences the tree partitioning to be deeper.

The advantage of this method is to predict the whole CTU tree partitioning features in one shot, which is very convenient to reduce the complexity overhead and latency introduced by this step. Furthermore, the MobileNetV2 architecture has been designed with a low number of operations in order to master the complexity of the inference stage.

8.2.3 Multi-Class Classifier

Following the CNN, the MCCs are designed based on DT models to predict a vector of split probabilities $\tilde{\mathbf{c}}$. MCCs exploit the features predicted by the CNN in order to have all the information of the tree partitioning inside the processed CU. Indeed, the input of the MCCs are the features related to the CU currently processed. Each CU size has its respective MCC in order to fit the number of input components and to improve the model accuracy by limiting the possibilities of split and to specialize each model to a fixed block size.

Table 8.1 – *Possible splits according to the CU size*

<table>
<tr><th>Height \ Width</th><th>128</th><th>64</th><th>32</th><th>16</th><th>8</th><th>4</th></tr>
<tr><td>128</td><td>QT, BT</td><td colspan="3">BT</td><td colspan="2">-</td></tr>
<tr><td>64</td><td rowspan="3">BT</td><td>All</td><td colspan="2">BT, TT</td><td rowspan="3">BT, TTH</td><td rowspan="3">BTH, TTH</td></tr>
<tr><td>32</td><td rowspan="2">BT, TT</td><td>All</td><td>BT, TT</td></tr>
<tr><td>16</td><td>BT, TT</td><td>All</td></tr>
<tr><td>8</td><td rowspan="2">-</td><td colspan="3">BT, TTV</td><td>BT</td><td>BTH</td></tr>
<tr><td>4</td><td colspan="3">BTV, TTV</td><td>BTV</td><td>-</td></tr>
</table>

TABLE 8.1 details the splits available depending on the CU size under RA configuration. The restriction on the splits are due to the limitation of the VTM encoder. For instance, TT split is not available when the width or the height is equal to 128. QT is not available when a BT or a TT split is previously performed.

Unlike in AI configuration, the RA configuration allows the CU size to start at the CTU size which is defined to be 128×128. Indeed, the VTM under RA configuration does not start the tree partitioning process with an implicit QT split. This table resumes the classes considered for each MCC according to the CU size.

Table 8.2 – *Sequences encoded to create our dataset selected from Xiph [110] and UVG [111] datasets.*

Sequence	Dataset	Width	Height	Sequence	Dataset	Width	Height
Aspen	Xiph [110]	1920	1080	Boxing practice	Xiph [110]	4096	2160
Beauty	UVG [111]	3840	2160	Narrator	Xiph [110]	4096	2160
Bosphorus	UVG [111]	3840	2160	Square and time lapse	Xiph [110]	4096	2160
Bus	Xiph [110]	352	288	Old town cross	Xiph [110]	3840	2160
City alley	UVG [111]	3840	2160	Paris	Xiph [110]	352	288
Coast guard	Xiph [110]	352	288	Race night	UVG [111]	3840	2160
Controlled burn	Xiph [110]	1920	1080	Ready set go	UVG [111]	3840	2160
Crowd run	Xiph [110]	3840	2160	Red kayak	Xiph [110]	1920	1080
Ducks take off	Xiph [110]	1920	1080	River bank	UVG [111]	3840	2160
Factory	Xiph [110]	1920	1080	Shake n dry	UVG [111]	3840	2160
Flower focus	UVG [111]	3840	2160	Sintel trailer	Xiph [110]	1920	1080
Flower kids	UVG [111]	3840	2160	Show mnt	Xiph [110]	1920	1080
Flower pan	UVG [111]	3840	2160	Stefan	Xiph [110]	352	288
Foreman	Xiph [110]	352	288	Stockholm	Xiph [110]	1280	720
Honey bee	UVG [111]	3840	2160	Sun bath	UVG [111]	3840	2160
In to tree	Xiph [110]	1280	720	Tempete	Xiph [110]	352	288
Jockey	UVG [111]	1920	1080	Tractor	Xiph [110]	1920	1080
Lips	UVG [111]	3840	2160	Twilight	UVG [111]	3840	2160
Mobcal	Xiph [110]	1280	720	Vidyo 4	Xiph [110]	1280	720
Mobile	Xiph [110]	352	288	Waterfall	Xiph [110]	352	288
Boat	Xiph [110]	4096	2160	Yacht ride	UVG [111]	3840	2160

8.3 Dataset

In RA configuration, the temporal relationship between frames has a major impact in the coding quality. As previously presented, our method exploits the temporal coding by taking advantage of the MVs computed for the CTU. The two optimal CTUs of reference B_{MV1} and B_{MV1} are a part of our input compared with our method in AI configuration where only the 68×68 block was needed. Then, our dataset is different from the previous one due to the new temporal prediction.

8.3.1 Dataset description

Instead of encoding independent frames, in RA configuration, full sequences are required to have a relevant dataset. Indeed, taking image dataset is no longer conceivable as the learning process of our ML techniques need to take into account the inter prediction.

TABLE 8.2 details the sequences considered for the training of the proposed method. Two video datasets were selected with UVG [111] and Xiph [110] to generate the dataset for the learning process. 26 sequences were taken from Xiph and 16 from UVG. With those sequences, several resolutions are provided from 352×288 to 4096×2160. All those sequences brought a wide variety of contents and resolutions.

To create the dataset and to generate the ground truth in order to train the CNN and DT models, the sequences are encoded with the VTM10.2 under RA configuration at QP 22, 27, 32 and 37. The four initial QPs lead to 16 different QPs from 23 to 46 depending on the initial QP selected and the POC of the processed frame. To limit the encoding time and the information redundancy under the same sequence, all the sequences are limited to the 32 first frames.

Figure 8.2 illustrates the dataset with its inputs and outputs. As presented earlier the CNN predicts the tree partitioning with $\mathbf{B}$ as input, the first dimension is the CTU

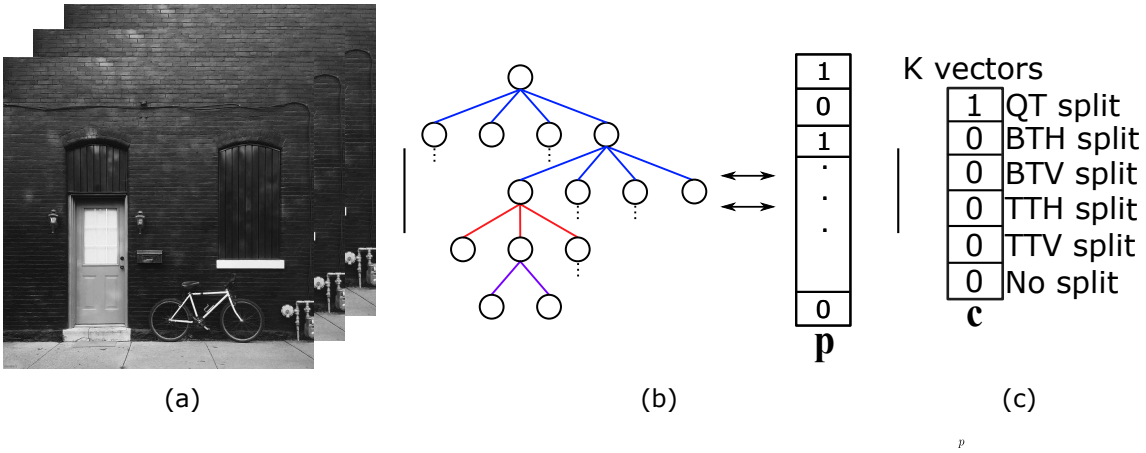


Figure 8.2 – *Representation of our dataset. (a) is the input block* **B**. *(b) is the optimal tree partitioning of the block represented by a tree and transformed to the soft representation, i.e., a 1984 elements vector* **p**. *(c) is the hard representation, K vectors* **c** *of 6 probabilities, that define the optimal split for each of the K blocks in the tree. K is the decision tree depth.*

currently processed B_{cur} and the second and third dimensions are the two 128×128 blocks of reference B_{MV1} and B_{MV2} relying on the MVs with the lowest RD-costs. B_{MV1} and B_{MV2} provide the temporal information easily without additional complexity in the VTM encoding process. The CNN and MCC ground truths are the tree partitioning of the CTU currently computed taken from the VTM exhaustive search.

The tree partitioning is saved as a tree that describes the different splits computed inside the CTU. Two representations are then required. The first one is the soft representation which is a vector **p** of 1984 elements that depicts the tree partitioning at each 4×4 edge of the CTU. The second one is the hard representation which details the tree partitioning through K vectors **c** containing 6 split probabilities that define the optimal split for each CU.

8.3.2 Dataset pre-processing

The dataset has an important impact on the performance during the learning process of our ML methods. The disparity of the dataset is one of the key aspect to maximize the performance especially its generalisation capability on unseen data. Thus, an analysis and an optimization on the disparity is conducted on the dataset described previously. First, the unbalanced dataset is separated into classes which represent the tree partitioning depths and the distribution of those classes is analysed. Then, the dataset is balanced and the distribution between the classes is exhibited.

Figure 8.3 details our unbalanced dataset on the left and the balanced dataset on the right which are both divided into classes. The classes go from 1 to 6 and represent the tree partitioning through no split to highly deep partition, respectively. Those classes are created relying on the sum of the vector **p** components. The sum intervals to create the classes are shown in the figure. The dataset where no filter is applied has a high disparity between the classes. Indeed, the class 0 which represents the CTU without any split contains more than half of the dataset. On the other hand, the class 6 which contains the deepest partitions represents 4% of the total dataset. 16 QPs are available inside this dataset which creates also an heterogeneous number of instances between the QPs. An equitable representation of the classes inside the dataset is a key aspect for the ML performance, so, a pre-processing

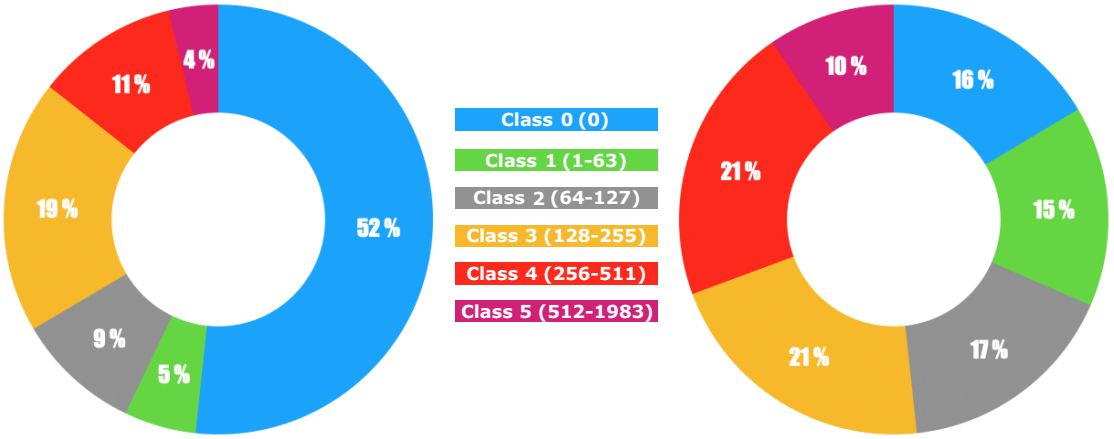


Figure 8.3 – *Breakdown of our inter prediction dataset by partition classes. (left) Unbalanced dataset. (right) Balanced dataset.*

is proposed to balance the dataset. The operation to balance the dataset first separates the dataset by QP categories. 18750 instances are taken for each QP value which results in 300 000 examples. Then, inside the QP category, the 6 classes are equally distributed with 3125 instances of each class. This leads to a dataset equally distributed over both QPs and classes. However, some QPs or classes have a lower number of instances. For instance, when a QP is high, the possibilities to obtain the class 6 is very low which leads to a higher proportion of no split. Nonetheless, the class 6 goes from 4% with the unbalanced dataset to 10% with the balanced dataset. Those optimizations are essential for the training process of our ML methods.

8.4 Training process and experimental setup

The balanced dataset detailed in the previous section is used for the training process of our ML models. Indeed the CNN and the MCC models are trained with the training set which contains 90% of the dataset. This section presents the CNN and MCC training processes with their respective frameworks and parameters. The experimental setup performed to obtain the complexity reduction and BD-BR on the VTM10.2 are also described.

8.4.1 Convolutional Neural Network training parameters

The dataset is augmented during the CNN training to increase its accuracy. Indeed, the dataset is a key point for the CNN performance, increasing the size of the dataset gives an higher number of instance to better optimize its weights. Data augmentation has been proceeded to increase the number of instance. The data augmentation relies on the instances taken from the VTM encoder. For one instance, three instances based on the first instance are created. The three instances correspond to a rotation in horizontal, vertical, and horizontal followed by vertical. One instance leads to four instances of the same partition but with different rotations. The rotations are performed on the input and the ground truth of the dataset, i.e., the CTU and its respective tree partitioning plus the two reference blocks.

Figure 8.4 presents the reference tree partitioning and its respective three flips. The reference is the tree partitioning obtained by encoding the sequence with the VTM. Then, an horizontal, vertical, and horizontal plus vertical flips based on the reference tree partitioning are proposed to augment the dataset.

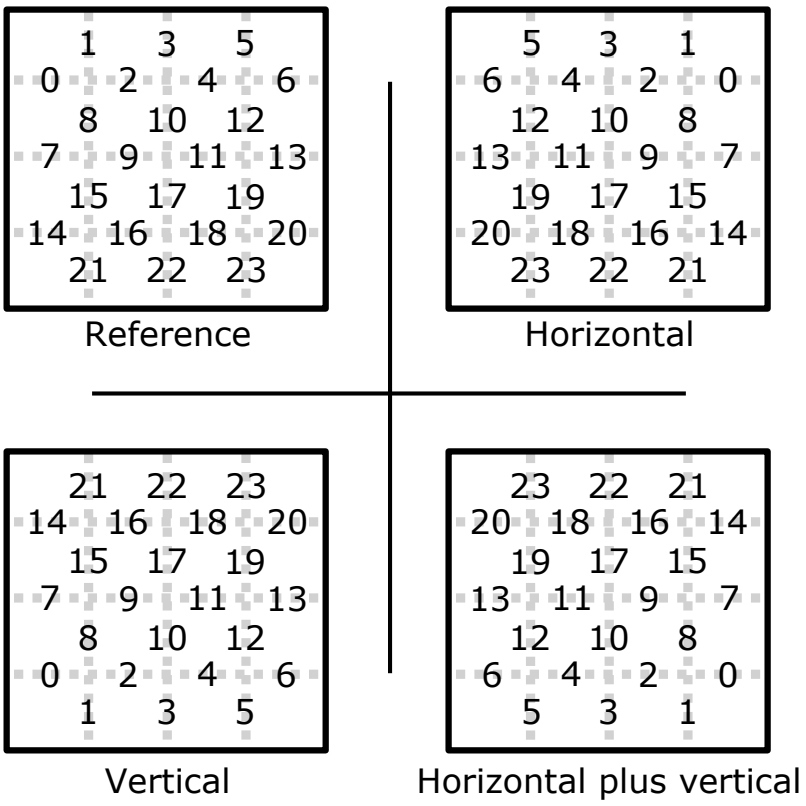


Figure 8.4 – *The process of data augmentation for the CNN training. In top-left the reference tree partitioning. In top-right the horizontal flip of the reference. In bottom-left the vertical flip of the reference. In bottom-right the horizontal plus vertical flip of the reference.*

The framework used to train the CNN is Keras [100] running on top of Tensorflow module [101]. The MobileNetV2 CNN weights are pre-trained on ImageNet. The pre-trained weights are then updated at each batch iteration with the ADAM stochastic gradient descent optimizer [102]. The batch size is set to 256 instances of one CTU and the learning rate is equal to 10^{-3}. The loss function optimizes the weights by minimizing the Mean-Squared Error (MSE) between the predicted probability vector $\tilde{\mathbf{p}}$ and its corresponding ground truth vector $\mathbf{p}$ as follows :

$$\mathcal{L}_{cnn} = \frac{1}{L}\sum_{l=1}^{L} ||\mathbf{p}^l - \tilde{\mathbf{p}}^l||_2^2. \tag{8.3}$$

A random shuffle of the dataset is applied at each one of the hundred epoch. Finally, the CNN training was carried out on a RTX 2080 Ti Graphics Processing Unit (GPU).

8.4.2 Multi-Class Classifier training parameters

The MCC is based on the LightGBM (LBGM) framework [108] version 2.3.1. LBGM is a gradient boosting framework based on decision tree developed by Microsoft. The LBGM MCC model is the gradient boosting decision tree method described in [108] which has a fast computation which is essential as the inference is performed for each CU. Indeed at each CU, the MCC determines the top split through the vector of probabilities. To compute the probabilities, the MCC takes the CNN output located at the CU currently processed. As presented in the previous chapter, the training of the MCC is optimized through the

minimization of the cross-entropy loss function defined as follows :

$$\mathcal{L}_{mcc} = -\frac{1}{L}\sum_{l=1}^{L} \mathbf{c}^l \cdot \log \tilde{\mathbf{c}}^l, \tag{8.4}$$

where $\mathbf{c}$ is the vector of ground truth split probabilities and $\tilde{\mathbf{c}}$ is the vector of split probabilities predicted by the model. Different parameters are defined with the number of boosting iterations and the early stopping process set at 100000 and 1500, respectively.

8.4.3 Experimental setup

The results are evaluated with predefined parameters to have a fair comparison with the state of the art methods. Indeed, the sequences, configurations, and parameters are fixed. The sequences selected are the Common Test Conditions (CTC) sequences, with the 7 classes going from A1 to F. This method reduces the complexity of the tree partitioning for inter coding so the selected coding configuration is the RA. The VTM10.2 is exploited to determine both complexity reduction and BD-BR performance.

Our solution is assessed in terms of both coding efficiency and computational complexity. The BD-BR [10] is computed to measure the coding efficiency over the four QPs 22, 27, 32, and 37 with the reference encoding and the encoding with our method applied. The complexity reduction is computed through the ΔET metric defined in Eq. 2.7 (page 16).

The proposed method is integrated in the VTM10.2 encoder describing the VVC encoding process with the C++ programming language. To integrate our CNN and MCC models which are trained under python, conversion need to be performed. For the CNN, the frugally-deep framework [103] is used to generate the inference source code in C++ programming language. The LBGM framework [108] allows the MCC to infer also in C++. Finally, all encodings were carried out sequentially on an Intel Xeon E5-2603 v4 processor running at 1.70 GHz under Ubuntu 16.04.5 operating system (OS).

8.5 Experimental results

Based on the experimental setup, this section details the results obtained by our proposed method. The results are described firstly for the CNN and the MCC in terms of loss and accuracy. And secondly, for the whole method implemented in the VTM 10.2 expressed through complexity reduction and BD-BR.

8.5.1 Convolutional Neural Network performance

As presented in Section 8.2, the CNN is the first part of our method. Its output is the basis of the method. Indeed, the features predicted by the CNN for each CTU are then exploited by the MCCs to define the most likely splits in the VTM. As it is the basis of our algorithm, it is mandatory that the CNN has the highest performance to enhance the information given to the MCCs.

Figure 8.5 details several CNNs and parameters with their respective losses from the training and validation parts. The training part is the set of instances that are used to train the CNN, whereas the validation part is the set of instances that are tested at each epoch but not used to learn the weights of the CNN. Four configurations are determined based on three CNNs with MobileNetV2 [109], an adaptation of ResNet [51], and EfficientNet [94]. Those CNNs have different applications, MobileNetV2 is a simple mobile architecture which

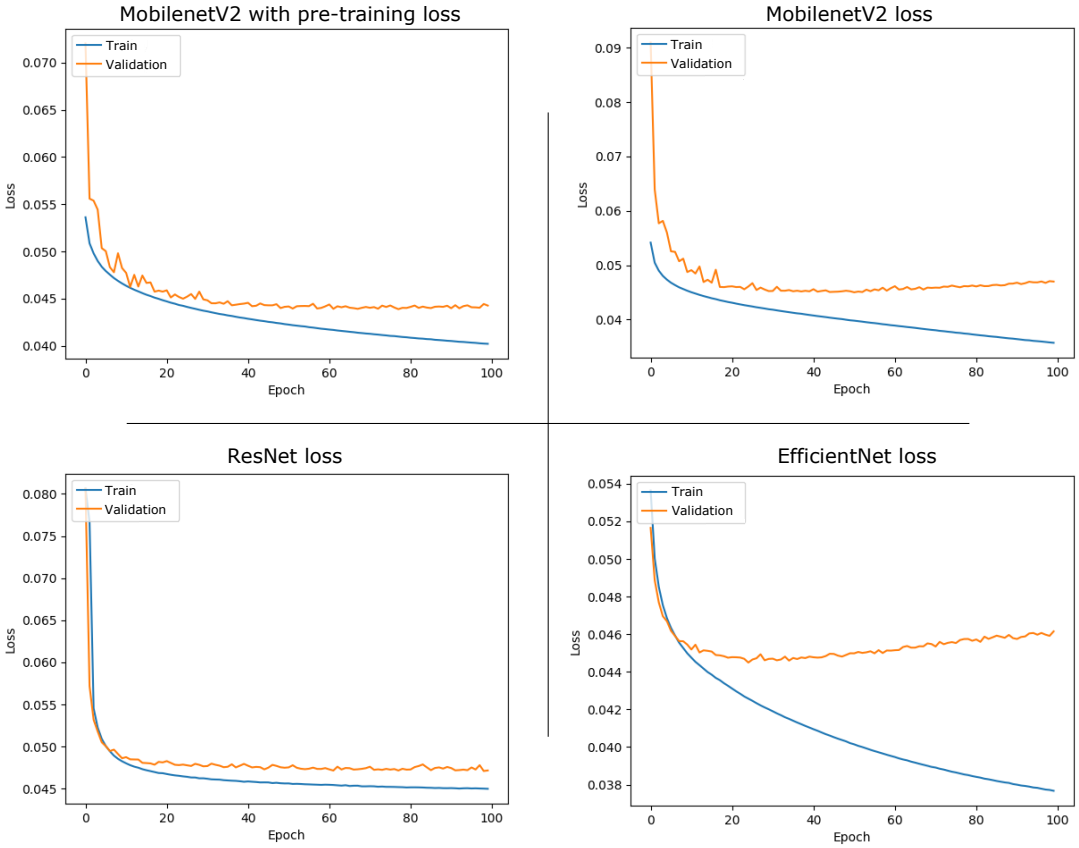


Figure 8.5 – *Several losses from the different CNNs trained with different parameters. Top-left, the MobileNetV2 loss with the weight pre-trained on imagenet. Top-right, the MobileNetV2 loss without pre-training. Bottom-left, the technicolor CNN loss. Bottom-right, the EfficientNet loss.*

is memory-efficient with 3.4M parameters. ResNet introduced the shortcut connection that skip layers, it has around 10M parameters for its version ResNet-18. Finally, EfficientNet is a deep network which goes from 5.3M to 66M parameters depending on the version that uniformly scales all dimensions. The EfficientNet version used is the EfficientNetB0 architecture with 5.3M parameters. MobileNetV2 with a pre-training on imageNet brings a loss of 0.04 under the training dataset and 0.044 under the validation dataset which is the best among the four tested configurations. The pre-training is significant to limit the over-fitting of the network. Indeed, without the pre-training, the result on the training set is better with a loss of 0.036, however, the validation set obtains a lower loss with 0.047. The CNN based on the ResNet architecture brings also interesting results with 0.045 for training and 0.047 for testing with negligible over-fitting. The difference of loss between the training part and the validation part of the EfficientNet network shows an over-fitting. This overfitting is due to the high number of parameters that are difficult to train with our dataset. Even if the network shows great performance under the training dataset, the validation dataset demonstrates that EfficientNet has lower results than MobileNetV2. MobileNetV2 with pre-training is selected due to its higher performance among the CNNs, furthermore, its accuracy is up to 94% under the validation set. This accuracy demonstrates that MobileNetV2 can predict 94% of good predictions from the vector, i.e., 94% of the predicted probabilities with a threshold of 0.5 are equal to the ground truth.

8.5.2 Multi-Class Classifier performance

Following the CNN, the MCCs based on DT models exploit the features to predict the split probabilities. For each CU size, a specialized MCC is created and optimized. Several MCCs are necessary as the number of features are adjusted based on the size of the considered CU. Indeed, the features have spatial locations inside the CTU, only the features involved in the considered CU are taken as inputs.

TABLE 8.3 details the results of the obtained MCCs through top-1 to top-3 accuracies. The models performance is separated according to the width and height of the processed CU. The number of splits available based on the CU size are also depicted in this table. On average, the top-1 accuracy is up to 78%, i.e., the MCC predicts on average the optimal split 8 times out of 10. The top-2 accuracy which defines that the optimal split is one of the two highest probabilities is up to 89%. Finally, on average, the MCC predictions reach 92% for the top-3 accuracy. With this accuracy and the number of splits available for each CU size, the complexity is almost halved with 9 out of 10 splits correctly predicted. More specifically, an average is proposed based on the number of classes available which is more relevant as it affects directly the accuracy performance. MCC with 2 classes (8×4 or 4×8 CU size) has, on average, 95% top-1 accuracy. The performance goes down to 54% for block with 6 available classes. When the number of classes increase, the prediction is harder as more split are available. This impact on the prediction performance is also reproduced for the top-2 and top-3 configurations. The top-2 accuracy goes from 98% to 73% for 3 to 6 available splits, respectively. The top-3 accuracy is performed on the 4 to 6 available classes with 96% to 85% accuracy, respectively. Separately, the accuracies are also provided for each MCC. The top-1 accuracies range from 96% to 45% for the blocks of size 64×4 and 32×32, respectively. Due to the dataset homogeneity, four models are achieving 100% top-1 accuracy. Indeed, our dataset generated for the MCC training has no diversity on specific sizes due to the VTM selection and only one split is available in the ground truth through all the instances for the sizes 128×32, 32×128, 128×16, and 16×128. The performance of the MCC optimizes the relevance of the features given by the CNN. Indeed, the optimal split is predicted by the MCC almost 80% of the time as the most probable split.

8.5.3 Complexity reduction and coding performance

The proposed model is integrated in the VTM10.2 under the RA configuration. As presented in the Chapter 3, the state of the art techniques proposed under the RA configuration obtain lower results compared with the techniques computed under the AI configuration. In order to compare our results relying on the configurations exposed further, two state of the art methods are selected. The first method proposed by Tang *et al.* [62] reduced the inter coding complexity through tree partitioning pruning. An handcrafted metric was defined to early terminate the unlikely splits. The second selected method is designed by Pan *et al.* [64] and reduced the complexity with binary classification deciding the split or the early termination. This binary classification is the output of a multi-information fusion CNN. This subsection presents the complexity reduction along with the BD-BR loss caused by our method and the two state of the art techniques.

TABLE 8.4 details three configurations used in our solution to reduce the VTM complexity. These configurations allow different complexity reduction-BD-BR trade-offs. Such as presented earlier, the top-N means that the N top split probabilities predicted by the MCCs are likely and that those splits will be performed. $C1$ computes the top-4 split predicted by the MCCs for all the block sizes. $C2$ performs the top-4 split for the blocks that

Table 8.3 – *Performance of every MCCs with their defined sizes and number of outputs through top-1, top-2, and top-3 accuracies on the validation set.*

Width	Height	Number of classes	Top-1 accuracy (%)	Top-2 accuracy (%)	Top-3 accuracy (%)
128	128	4	69.41	87.78	98.05
128	64	3	85.34	95.22	-
64	128	3	84.53	93.45	-
128	32	3	100	100	-
32	128	3	100	100	-
128	16	3	100	100	-
16	128	3	100	100	-
64	64	6	61.46	77.69	88.16
64	32	5	54.98	74.69	87.41
32	64	5	66.27	81.93	91.22
64	16	5	55.61	77.09	89.73
16	64	5	68.71	83.87	92.93
64	8	4	71.94	86.48	95.67
8	64	4	71.44	86.61	95.55
64	4	3	96.36	98.77	-
4	64	3	94.53	98.60	-
32	32	6	45.25	67.12	81.50
32	16	5	60.55	77.53	89.40
16	32	5	59.30	77.15	89.24
32	8	4	67.26	85.59	95.32
8	32	4	62.47	82.66	94.65
32	4	3	80.84	93.27	-
4	32	3	82.82	96.14	-
16	16	6	54.87	73.95	86.73
16	8	4	83.58	92.95	97.65
8	16	4	80.79	91.75	97.29
16	4	3	92.89	98.04	-
4	16	3	92.50	98.25	-
8	8	3	91.39	96.73	-
8	4	2	95.27	-	-
4	8	2	94.53	-	-
Average		2	94.9	-	-
		3	92.4	97.57	-
		4	72.41	87.69	96.31
		5	60.90	78.71	89.99
		6	53.86	72.92	85.46
		-	78.22	88.73	91.91

Table 8.4 – *The configurations performed in the VTM depending on the number of split performed.*

Configuration	Block size					
	128	64	32	16	8	4
$C1$	top-4	top-4	top-4	top-4	top-4	top-4
$C2$	top-4	top-4	top-3	top-3	top-3	top-3
$C3$	top-3	top-3	top-3	top-3	top-3	top-3

have the width or the height higher or equal than 64 and the top-3 for the other blocks. $C3$ computes only the top-3 split for each block size. Furthermore, another result is proposed relying on a different configuration. Indeed, to determine if the two reference CTUs B_{MV1} and B_{MV2} are useful, a CNN with only the considered CTU is assessed.

TABLE 8.5 depicts our results compared with two state of the art methods [62], [64]. All the results are provided on the CTC sequences, furthermore, an average is computed on each class from class A1 to class F and on all classes except class F. In Tang *et al.* [62] work, no results were given on classes A1, A2, and some sequences of class B and F. In Pan *et al.* [64] work, no results were given for the class F which contains specific sequences.

The configurations of our final method are depicted in the table from $C1$ to $C3$. In average, our $C1$ configuration reaches the lowest complexity reduction with 31.8% for 1.11% BD-BR loss. The $C2$ configuration brings 43.4% complexity reduction for 2.33% BD-BR loss. Finally, the highest complexity reduction is achieved by the $C3$ configuration with 54.3% for a BD-BR loss of 3.12%.

In comparison with our method without the two reference CTUs, the $C2$ configuration obtains the most similar results across the CTC classes. For high resolution classes, the $C2$ configuration obtains better performance in terms of both complexity reduction and BD-BR loss. For instance, with class A1, the $C2$ configuration has a gain of 4.2% complexity reduction for the same BD-BR loss. With class A2, $C2$ obtains a gain of 1.5% complexity reduction in addition with almost 0.5% BD-BR gain. However, under low resolution (classes C and D) and specific content (class F) classes, our method without the two reference CTUs, B_{MV1} and B_{MV2}, obtains better results. One of the reasons is the size of the CTU compared with the sequence. Indeed, under low resolution sequences, a CTU takes an important proportion of the frame. To improve our method under low resolution sequences, the MV can be computed on smaller block sizes in order to take into account the diversity of the pixels movements.

Compared with the two state of the art methods [62], [64], our lowest complexity reduction performance with the $C1$ configuration has higher complexity reduction for lower BD-BR loss. Indeed, the $C1$ configuration has a gain of 0.19% and 1.41% BD-BR loss compared to [62] and [64], respectively. In terms of complexity reduction, our $C1$ configuration obtains a gain of 2.3% and 6.5% compared with [62] and [64], respectively. With approximately the same BD-BR loss our $C2$ configuration increases the complexity reduction by 20% compared with [64].

Moreover, in the table, the performance is given for sequences and classes. The results when comparing the classes demonstrate that the best trade-off between complexity reduction and BD-BR loss is the class A1 with 51.1% complexity reduction for 1.81% BD-BR loss which compared with [64] has 15.4% more complexity reduction with 0.88% less BD-BR loss. As shown in the table, the high resolution classes has better results than the lowest one. For instance, the class D which has the lowest resolution obtains with the $C1$ configuration 21.4% complexity reduction for 1.04% BD-BR loss. This is due to the dataset which is not equally distributed in terms of resolutions. Indeed, having the same number of CTU from the resolution 416×240 than 3840×2160 is an issue as one frame from 416×240 resolution contains 8 CTUs compared to 510 CTUs from a 4K frame. 64 sequences with resolution 416×240 are required for one 4K sequence to fill the important gap between the two resolutions in terms of CTU instances.

The best performance through the sequences is obtained by FoodMarket4 with 55.3% complexity reduction for 1.18% BD-BR loss. Compared with [64], our method brings 12.4% more complexity reduction for 0.41% BD-BR gain.

TABLE 8.5 – *ΔET and BD-BR performance of the proposed solution in comparison with state of the art techniques in RA coding configuration.*

Class	Sequence	Tang *et al.* [62] VTM 4.0.1rc1		Pan *et al.* [64] VTM 6.0		Ours without MVs VTM10.2		Ours $C1$ VTM10.2		Ours $C2$ VTM10.2		Ours $C3$ VTM10.2	
		BD-BR	ΔET	BD-BR	ΔET	BD-BR	ΔET	BD-BR	ΔET	BD-BR	ΔET	BD-BR	ΔET
A1	Tango2	-	-	3.68%	34%	2.13%	42.7%	1.16%	40.1%	1.48%	47.7%	3.38%	60.7%
	FoodMarket4	-	-	1.59%	42.9%	1.03%	39.8%	1.11%	52.2%	1.18%	55.3%	2.58%	67.9%
	Campfire	-	-	2.8%	30.1%	2.27%	58.4%	1.10%	33.3%	2.78%	50.3%	3.63%	62.8%
	Average	-	-	2.69%	35.7%	1.81%	46.9%	1.12%	41.9%	1.81%	51.1%	3.20%	63.8%
A2	CatRobot1	-	-	5.59%	30.6%	2.38%	39.3%	0.99%	35.5%	1.50%	43.9%	3.11%	55.5%
	DaylightRoad2	-	-	4.43%	29.2%	2.61%	40.6%	1.36%	37.5%	2.53%	45%	3.77%	55.9%
	ParkRunning3	-	-	1.61%	21.3%	1.90%	49.3%	0.66%	31.2%	1.54%	44.9%	1.85%	56.5%
	Average	-	-	3.88%	27%	2.30%	43.1%	1%	34.7%	1.86%	44.6%	2.91%	55.9%
B	MarketPlace	-	-	3.22%	36.5%	2.13%	44%	1.01%	35.6%	1.90%	45.1%	2.83%	57.1%
	RitualDance	-	-	2.97%	31.2%	2.95%	51.1%	1.47%	39.3%	2.84%	50.3%	4.23%	62.2%
	Cactus	1.51%	33.2%	5.2%	25.4%	2.08%	49.1%	0.87%	33.7%	1.90%	45.5%	2.60%	57.6%
	BasketballDrive	2.69%	42.1%	2.96%	32.4%	3.05%	54.1%	1.61%	39.3%	2.60%	50.6%	3.92%	62.9%
	BQTerrace	1.19%	29.5%	0.98%	13.8%	1.60%	41%	0.75%	29.1%	1.80%	41.2%	1.87%	51.3%
	Average	1.80%	34.9%	3.07%	27.9%	2.36%	47.9%	1.14%	35.4%	2.21%	46.5%	3.09%	58.2%
C	RaceHorses	2.96%	33.9%	2.23%	22.5%	3.01%	53.2%	1.41%	28%	3.22%	44.8%	3.96%	54.6%
	BQMall	1.49%	33.3%	2.35%	22.4%	3.43%	50.3%	1.14%	27.7%	3.19%	41.8%	3.98%	53.2%
	PartyScene	1.05%	35.2%	1.84%	14.9%	2.07%	48.9%	1.63%	22.2%	3.51%	41.1%	3.78%	50.1%
	BasketballDrill	1.36%	28.7%	1.59%	24.4%	2.52%	53%	1.22%	31.9%	2.88%	44.9%	3.44%	57.2%
	Average	1.72%	32.8%	2%	21%	2.76%	51.3%	1.35%	27.5%	3.20%	43.1%	3.79%	53.8%
D	RaceHorses	1.59%	31.8%	2.24%	20.3%	3.25%	48.3%	0.96%	21.2%	3.40%	41.2%	4.28%	49%
	BQSquare	-0.11%	23%	0.84%	9.7%	2.16%	36.9%	1.24%	17.6%	2.58%	30.3%	2.18%	39.1%
	BlowingBubbles	0.73%	21.9%	2.29%	17%	2.10%	46.5%	0.67%	24.1%	2.44%	39.5%	2.97%	48.1%
	BasketballPass	-0.61%	24.3%	1.56%	21.2%	2.53%	43.2%	1.28%	22.5%	3.68%	36%	3.64%	44.6%
	Average	0.4%	25.2%	1.73%	17%	2.51%	43.7%	1.04%	21.4%	3.02%	36.8%	3.26%	45.2%
E	FourPeople	1.37%	26.6%	1.76%	25.3%	1.40%	34.6%	0.68%	31.3%	1.48%	38.3%	1.97%	48.9%
	Johnny	1.51%	24.4%	1.69%	24.9%	1.38%	30.2%	0.97%	32.2%	1.23%	37.5%	2.45%	48.6%
	KristenAndSara	1.53%	25.3%	2.11%	26.2%	1.62%	32.9%	1.09%	33.6%	1.65%	40.3%	2.17%	51.2%
	Average	1.47%	25.4%	1.85%	25.5%	1.47%	32.6%	0.91%	32.3%	1.45%	38.7%	2.20%	49.6%
Av.		1.30%	29.5%	2.52%	25.3%	2.25%	44.9%	1.11%	31.8%	2.33%	43.4%	3.12%	54.3%
F	ArenaOfValor	-	-	-	-	2.67%	50%	1.44%	33.2%	3.43%	43.8%	4.40%	55.2%
	BasketballDrillText	1.52%	31.2%	-	-	2.54%	53.6%	1.39%	32.4%	3.30%	44.7%	3.88%	57.5%
	SlideEditing	0.55%	29.3%	-	-	1.50%	25.7%	1.67%	28.7%	2.17%	34.4%	2.29%	42.1%
	SlideShow	0.98%	37.7%	-	-	1.22%	34.2%	2.09%	31%	2.95%	38%	4.58%	49.2%
	Average	1.02%	32.7%	-	-	1.98%	40.9%	1.65%	31.3%	2.96%	40.2%	3.79%	51%

Class F is composed of specific content sequences such as screen content category which does not appear in our dataset as it focuses on natural scenes. On average, our method has lower results than [62] with 1.65% BD-BR loss for 31.3% complexity reduction. However, [62] does not provide the results of the sequence ArenaOfValor. For the two sequences containing slides, [62] has better results with less than 1% BD-BR loss and more than 29% complexity reduction. This is also mainly caused by the diversity of our dataset. For BasketballDrillText which is the most similar to natural sequences inside the class F, our solution in both BD-BR and complexity reduction, has better results than [62] with 1.2% more complexity reduction and 0.13% BD-BR gain.

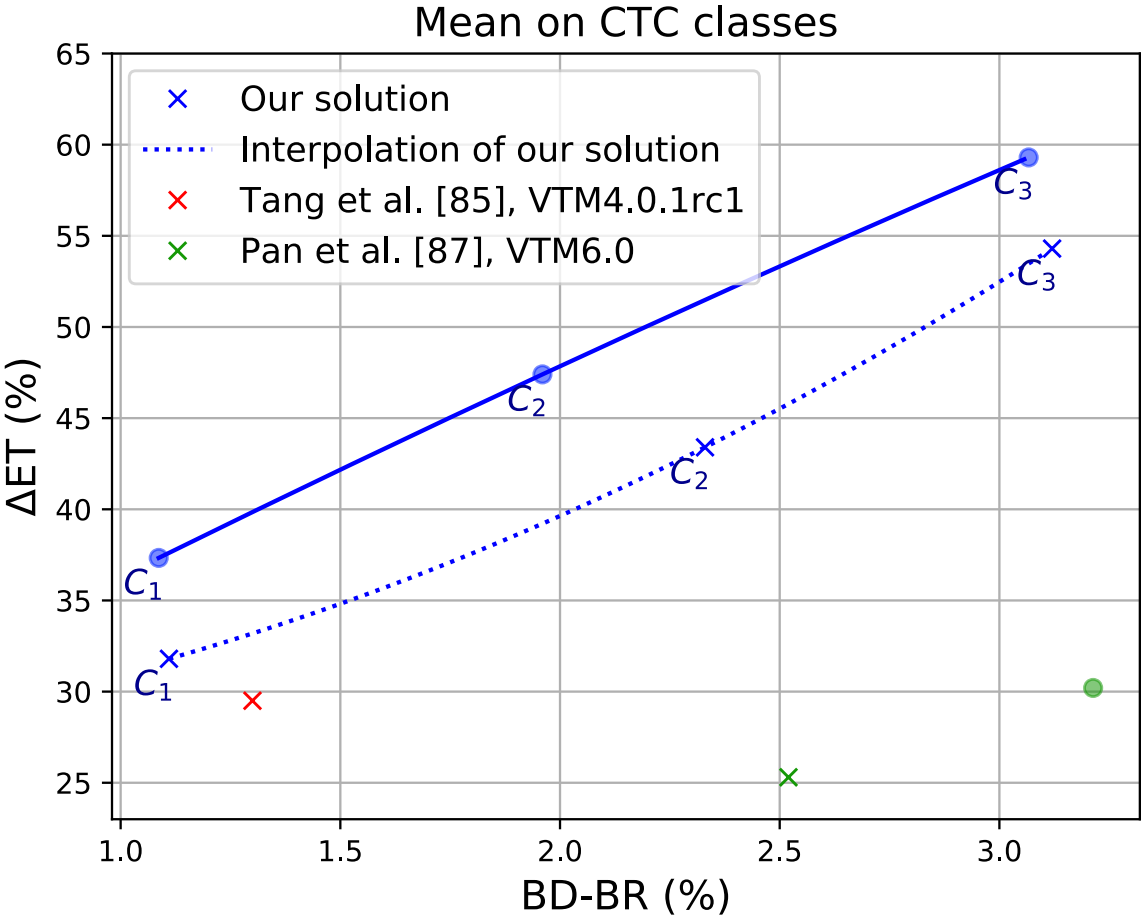


Figure 8.6 – *Performance comparison between the proposed solution through different configurations and the state of the art techniques. Circles are average results of Full Hd and 4K sequences. Crosses are an average through the CTC classes without class F.*

Figure 8.6 recaps the performance obtained by our method compared with state of the art techniques [62], [64] in terms of both BD-BR loss and complexity reduction. As shown in this figure, our results have better BD-BR-complexity reduction trade-offs than state of the art techniques. Furthermore, on Full HD and 4K sequences, the results obtained by our solution are better than the average on all the classes in terms of both BD-BR and complexity reduction. For instance, the $C2$ configuration on Full HD and 4K sequences has an increase of 4% in complexity reduction and a BD-BR gain of 0.37% compared with the average on the CTC.

8.6 Conclusion

In this chapter, a two-stages ML method is proposed to reduce the complexity of the VTM encoder under inter coding while limiting the BD-BR loss. This solution takes advantage of the MV computed before the tree partitioning process. First, the CNN takes as input the current CTU B_{cur} plus its two reference CTUs, B_{MV1} and B_{MV2}, obtained by the two MVs with the lowest RD-costs. Then, for each CU, a MCC predicts the split probabilities based on the features defined by the CNN. Finally, several configurations are provided to obtain different trade-offs between complexity reduction and BD-BR loss.

Our $C1$ solution brings 31.8% complexity reduction for 1.11% BD-BR loss on the CTC. This result surpasses in both complexity reduction and BD-BR loss the two state of the art techniques. The results on the CTC classes and on the Full HD and 4K sequences obtain different complexity reduction and BD-BR loss. Indeed, the high resolution sequences achieve higher complexity reduction for a lower BD-BR loss compared with the low resolution sequences. To limit this gap between the resolutions, the dataset heterogeneity is an important factor as the lowest resolution contains a negligible number of CTUs. Another solution is to create several CNNs based on the resolution. Furthermore, to improve our global results, a possibility is to focus on the information given to the MCC process and especially the tree partitioning of the two reference CTUs.

Troisième partie

Conclusion

CHAPITRE 9

Conclusion

During the standardisation process, the Versatile Video Coding (VVC) standard introduced several new tools that enhance its coding efficiency. Indeed, VVC brought a significant improvement in terms of coding efficiency over the previous standard High Efficiency Video Coding (HEVC). VVC provided a bitrate savings of 37% over HEVC in inter coding at the expense of a significant increased in encoder computational complexity. This complexity increase was caused by the new tools included in VVC and particularly the block partitioning process. The bitrate savings will limit the storage expansion and reduce the transmission bandwidth of video content. However, this huge complexity increase raised an issue for real-time encoding which is a key aspect for the development of this VVC standard.

In this context, this thesis presented four contributions in order to analyse and limit the encoding complexity. The first contribution determined that the tree partitioning is the tool with the most impact on the encoding complexity. The three other contributions were then dedicated to limit the complexity of the tree partitioning process. These contributions aimed to reduce the encoder complexity while minimizing the coding quality loss.

This chapter concludes this thesis through a summary of the contributions conducted during this thesis and the future works proposed to enhance our work.

9.1 Complexity reduction opportunities in the practical Versatile Video Coding encoder

9.1.1 Key results

In Chapter 4 the VVC Test Model (VTM) complexity has been studied through different aspects. First, the complexity has been assessed based on the video resolution, the Quantization Parameter (QP), and the splits. The video resolution has an important impact on the complexity as the complexity is related to the number of pixels. However, the video content has also an influence on the complexity as the VTM contains early termination process. Such as the video resolution, the QP is an important factor for complexity, i.e., the complexity increases when lowering the QP. Finally, the new splits Binary-Tree (BT) and Ternary-Tree (TT) which have been introduced in VVC brought a significant increase in complexity. In both All Intra (AI) and Random Access (RA) configurations, the BT split has the most impact on the encoding quality and on the complexity. Secondly, the com-

plexity opportunities were evaluated under AI and RA configurations. The tree partitioning was shown to have the highest potential for complexity reduction, up to 97%, whereas the respective percentages for intra mode prediction and Multiple Transform Selection (MTS) were 65% and 55% with VTM3.0 under AI configuration. Complexity reduction opportunity for the VTM10.2 shared the same result than the one obtained for the VTM3.0 under AI configuration for the tree partitioning process. Under RA configuration for the tree partitioning process, the complexity reduction opportunity reached 91.2% which is closed to the AI opportunity.

9.2 Threshold based partitioning scheme for Versatile Video Coding intra encoders

9.2.1 Key results

Based on the previous chapter, the tree partitioning tool has been selected to reduce the complexity of the VTM encoder. This first contribution presented in Chapter 6 exploited a Convolutional Neural Network (CNN) detailed in Chapter 5 that predicts the 64×64 block partition. The luminance block pixel values plus additional lines on the top and the left of the 64×64 block that gather the intra mode information were given as input to the CNN. Relying on these 68×68 block pixel values, the CNN predicted a vector of 480 probabilities. These probabilities corresponded to the 4×4 block boundaries of the 64×64 block partitioning. Then, by taking advantage of the vector probabilities, the probability of each split were determined. In order to skip unlikely splits, the split probabilities were compared with thresholds. Two solutions were designed to establish the thresholds. The first one was based on a uniform threshold, i.e., the same threshold value for the different classifiers, that was determined relying on the targetted trade-off between complexity reduction and Bjøntegaard Delta Bit Rate (BD-BR) loss. The second one optimized the thresholds based on the split impact on both complexity reduction and BD-BR. The thresholds were determined non-uniformly depending on the size of the Coding Unit (CU) and on the considered split. With the VTM6.1 under AI configuration, the uniform threshold solution enabled 42.2% complexity reduction for a slight BD-BR increase of 0.75%. The non-uniform threshold solution reached 54.5% complexity reduction for 1.33% BD-BR loss.

9.2.2 Future works

This contribution, presented in the Chapter 6, is a first solution to reduce the complexity through CNN prediction. In order to skip unlikely splits, split probabilities were compared with thresholds that were set uniformly and non-uniformly based on the BD-BR-complexity reduction trade-off. Different aspects of this solution can be enhanced. Firstly, the split probability computation can be changed to better exploit all the probabilities given by the CNN. For instance, instead of computing the split probabilities based on their locations, a matrix with different weights can be used to take advantage of the neighbour probabilities in order to better fit the split probability. Secondly, the non-uniform threshold were selected relying only on the split category and the block size. To define the final threshold for each of these parameters, an algorithm selects the highest trade-off while limiting the BD-BR loss. However, the trade-offs were computed alone for each split category and block size, the cumulative of two thresholds increase will not be a perfect addition of their BD-BR losses and their complexity reductions. To enhance the algorithm selection, a parameter

can be introduced to increase or reduce the BD-BR or complexity reduction impact for the same split or the same size. Furthermore, the thresholds can be optimized even further by creating categories such as the resolution or categories for sequence content.

9.3 Machine learning based Quad-Tree-Multi-Type Tree partitioning scheme for Versatile Video Coding intra encoders

9.3.1 Key results

The previous Chapter 6 skips unlikely splits based only on the location of the splits within the CNN output. This contribution, described in Chapter 7, takes advantage of all the vector probabilities inside the processed block. Based on the 68×68 block pixel values, the CNN outputted the 480 vector components. Then, the vector probabilities were exploited by the Decision Tree (DT) models to take advantage of all the probabilities inside the block computed instead of only the spatial location of a specific split. A DT model was trained for each block size in order to predict the split probabilities. Finally, different trade-offs were obtained from the top-N configuration selected. Concerning the top-3 configuration, our proposal assessed on the VTM under AI configuration enabled on average 46.6% complexity reduction for a negligible BD-BR loss of 0.86%. The top-2 configuration was able to reach a higher complexity reduction of 69.8% for 2.57% BD-BR loss. Compared with the previous contribution, for the same complexity reduction, a BD-BR gain of 1% was obtained. This gain was obtained through the usage of the whole CNN output information instead of just the probabilities at the spatial location of the considered split.

9.3.2 Future works

This contribution exploits the CNN prediction through a set of classifiers based on DTs that predict the split probabilities. Only the probability at the location of the split was taken into account to skip unlikely splits in Chapter 6 whereas in this chapter a DT was trained for each block size. Then, this DT takes as input the whole probabilities inside the computed block. However, other features can be injected inside the DT to optimise its performance. For instance, classical features used in Machine Learning (ML) related to image such as variance or gradient will add information of the block in order to better determine the split probabilities. Furthermore, a study on the feature importance is interesting in order to limit the number of input without loosing in accuracy. With this limitation on the number of input, we can think of gathering multiple DTs in one in order to reduce the memory taken by the DT weights.

9.4 Machine learning based Quad-Tree-Multi-Type Tree partitioning for Versatile Video Coding inter coding

9.4.1 Key results

The two previous contributions were aiming at reducing the VTM complexity under AI configuration. This last contribution described in Chapter 8 focused on reducing the complexity of the RA configuration by exploiting ML techniques. First, a CNN takes advantage of the inter information to predict the whole Coding Tree Unit (CTU) partitioning. Indeed, the two best Motion Vector (MV)s were calculated inside the VTM process before the tree

partitioning process. Then, the pixel values of the current CTU plus the two CTUs retrieved by the two best MVs were provided as input to the CNN to predict the vector that contained the probabilities of the 4×4 block boundaries of the whole CTU. Finally, the DT models were taking advantage of the probabilities inside the CU computed to predict the split probabilities. Based on the selected top-N configuration, different trade-offs were obtained in terms of both complexity reduction and BD-BR loss. On average, under the VTM10.2 in RA encoding, our $C1$ configuration reached the lowest complexity reduction with 31.8% for 1.11% BD-BR loss. The highest complexity reduction was obtained by the $C3$ configuration with 54.3% for a BD-BR loss of 3.12%. Our solution to reduce the complexity of inter coding obtained lower performance than our solutions under intra coding. As the frames in inter coding compared to intra coding were encoded with a lower number of bits for the same quality each misclassification have an important impact. Indeed, this lower performance was due to the higher impact on the BD-BR when skipping the optimal split in inter than in intra coding. However, more information is available in inter coding than in intra coding that can be exploited to better predict the partition tree. Those features will be presented thereafter.

9.4.2 Future works

The final contribution detailed in the Chapter 8 has reduced the complexity of the inter coding. Our solution took advantage of the optimal MVs computed by the VTM for the CTU size. This gave the information of the motion field between the CTU computed and its reference through the classical MV which is not always the optimal prediction. Indeed, intra mode prediction or the new inter mode prediction such as the affine motion compensated prediction can be selected. In order to better match the prediction information, it is possible to use the optimal inter prediction information which can be performed by another tools than the classical MV. This new optimal CTU of reference can be provided as input to the CNN. The intra mode prediction can also be the optimal tool selected under the RA configuration. We can estimate if the block computed is predicted by the intra or the inter mode through ML techniques or directly through the VTM Rate-Distortion (RD)-costs. Relying on this estimation, the optimal tree partitioning prediction method can be chosen to better fit to the selected mode.

Furthermore, the CTU gave the overall information of the motion field which is not accurate. Indeed, the VTM can split the CTU into smaller blocks. Those smaller blocks will have a different MV than those of CTU. In order to have more accurate motion information, a pre-processing is necessary. This pre-processing will serve to obtain the MV of all the 4×4 blocks included in the CTU currently computed. Instead of having the optimal CTU of reference, this technique will reconstruct a CTU of reference with the 4×4 blocks of reference which increases the accuracy for inter information.

9.5 Other perspectives

To conclude, some thoughts are proposed to enhance globally our contributions and their benefits in this domain.

We have shown in our three contributions that the high resolutions sequences obtain better results in terms of both BD-BR loss and complexity reduction. This highlights the impact of the ML training and especially the dataset influence. Indeed, the dataset was generated mostly with high resolutions sequences or images datasets. In order to resolve this issue and achieve higher performance on lower resolutions, several solutions are relevant.

One the them is to enhance the dataset heterogeneity by adding more low resolutions sequences. However, for one frame at 480p or 4K resolution, the sample of CTU obtains are 24 and 510, respectively, i.e., to achieve the same number of CTU instance from one frame of 4K resolution we need more than 21 frames of 480p resolution. This solution is challenging to implement as video dataset are not always available publicly. Another solution to reach the same performance on several resolutions are to separate the training of the ML methods by resolution categories. This will enhance the ML performance as it specialize the learning but at the cost of a memory increase by storing the several ML methods needed for the resolution categories.

Furthermore, our work is only performed under the VTM which is the VVC standard reference software. This software is implemented to evaluate the tools introduced and proposed during the VVC standardisation. Those tools are introduced in the VTM without much optimisation in terms of complexity. This leads the VTM to be highly expensive in terms of complexity. Our methods aim at reducing the encoders complexity through its tree partitioning process. Implementation under real-time encoders will be a different challenge than reducing the VTM complexity. Indeed, our contributions included in the VTM targetted to significantly reduces its complexity while maintaining the BD-BR gain. The real-time encoders could have a distinct goal which is to enhance the BD-BR while maintaining the complexity.

Finally, in this thesis classical ML techniques were considered to reduce the encoding complexity. Indeed, CNNs and DTs were used to predict the split probabilities in order to skip unlikely splits. However, during the last decades the ML domain has grown into significantly. Enhancing the performance of our prediction to better match the optimal partition is a key aspect of our methods. Several ways of enhancing the prediction accuracy are available. Indeed, the accuracy can be increased through the training parameters, the dataset heterogeneity, or the CNN structure. All these directions can be conducted and compared with an exhaustive search to obtain the highest accuracy. Furthermore, several newly developed ML techniques are an opportunity for enhancing those performance. For instance, the reinforcement learning which optimises its performance through rewards could be an interesting feature to exploit. Compared with the CNN that optimises its weights based on a predefined partition, the reinforcement learning can optimises its weights based on the RD-cost which is more reliable. Indeed, trying to copy the partition choice is not optimal as early termination processes are implemented in the VTM. Furthermore, the VTM only minimises the RD-cost on the actual block and can not exploit the whole frame or the whole sequence to optimise its selections. The reinforcement learning is able to exploit the whole frame or the whole sequence information as the environment to deduce the partition of each CTU.

ANNEXE A

French summary

Les possibilités d'exploitation des applications médias ont été révolutionnées au cours de la dernière décennie avec l'émergence de nouveaux services tels que la vidéo à la demande, le streaming vidéo ou les plateformes de partage de vidéos. Par ailleurs, les innovations apportées au domaine de la communication de données ont également transformé la quantité de données transmises. En effet, les technologies 4G, 5G, et la fibre optique ont permis une augmentation significative de la bande passante et sont donc plus à même de faire face à l'explosion des données vidéo.

La vidéo représente la grande majorité du trafic mondial de données selon l'étude de Cisco [1] datant de 2019. L'évolution du trafic IP mondial à travers les différentes catégories d'applications est représentée sur la Figure 1.1 de 2017 à 2022. En 2017, le trafic IP mondial était supérieur à 100 exaoctets par mois et Cisco prévoit que le trafic IP atteindra près de 400 exaoctets par mois en 2022. Le montant total du trafic IP mondial a presque quadruplé en 5 ans.

La catégorie ayant le plus d'impact sur le trafic IP mondial est la vidéo sur internet avec plus de 60 exaoctets par mois en 2017 et devrait atteindre 275 exaoctets par mois en 2022. La vidéo sur internet a pris plus de la moitié de l'ensemble du trafic IP en 2017 et continuera de croître, avec 71% en 2022. En outre, la vidéo à la demande sur Internet représente également une part importante de l'ensemble du trafic IP, de 20% en 2017 à 11% en 2022. En moyenne, 82% du trafic IP total (315 exaoctets par mois) est consacré à la transmission vidéo.

Les formats vidéo émergents tels que la 8K, le High-frame rate (HFR), la vidéo 360°, le multi-view, le light field et le point clouds augmentent la qualité de l'expérience par la résolution, la fréquence d'images ou l'immersion mais au prix d'une augmentation significative de la bande passante. Par exemple, une image 8K Ultra High Definition (UHD) contient 33 177 600 pixels, soit 96 fois plus que l'image Standard Definition (SD) (720×480). Plus précisément, la Figure 1.2 détaille le trafic vidéo sur internet de 2017 à 2022 divisé en trois résolutions : SD, High Definition (HD) et UHD. En 2017, le trafic vidéo sur Internet était partagé entre les résolutions SD et HD. En 2022, il est prévu que la résolution UHD représente 22% du trafic vidéo mondial et que la résolution SD chute à 21%. La résolution HD conserve la même proportion entre les catégories mais les données vidéo transmises sont quadruplées entre 2017 et 2022.

Les nouveaux formats vidéo et l'explosion du trafic vidéo IP nécessitent de nouvelles techniques de codage vidéo plus performantes que les techniques existantes. Des organi-

sations telles que l'ISO/IEC, l'UTU-T et Alliance for Open Media (AOM) définissent de nouvelles normes de codage vidéo. Plus récemment, AOM a développé le codec AV1 publié en 2018 comme successeur de VP9 et Joint Video Expert Team (JVET) a normalisé VVC ITU-T H.266 | MPEG-I - Partie 3 (ISO/IEC 23090-3) [2] en juillet 2020 comme successeur de HEVC.

A.1 Challenges et objectifs

L'objectif du codage vidéo est de minimiser un compromis entre débit et distorsion, appelé coût RD, c'est-à-dire d'obtenir le débit minimal pour une qualité donnée ou la qualité maximale pour un débit donné. Toutes les normes internationales de codage vidéo partagent la même structure de codage vidéo hybride. Par conséquent, au cours de la normalisation, différents outils de codage ont été intégrés pour améliorer les prédictions intra et inter, les transformées ou le processus de partitionnement des blocs. La sélection d'outils de codage spécifiques conduit à des efficacités de codage et des coûts de calcul différents. Des études comparatives ont été menées entre VVC et AV1 [3], [4] sur la base de mesures d'évaluation de la qualité subjective et objective, notamment Peak Signal-to-Noise Ratio (Psnr) et Video Multimethod Assessment Fusion (VMAF). VMAF est une métrique de qualité basée sur le ML qui s'appuie sur la métrique de perte de détails, la mesure de fidélité de l'information visuelle et la différence temporelle moyenne entre les images. Les deux travaux concluent que VVC surpasse AV1 en termes de scores de qualité objectif et subjectif. Cependant, le coût de calcul de AV1 est légèrement inférieur à celui de VVC et varie selon les configurations de codage.

Le logiciel de référence de VVC, appelé VTM, met en œuvre tous les outils de codage normatifs VVC permettant l'évaluation du débit-distortion et de la complexité ainsi que les tests de conformité. Le VTM apporte des réductions de débit de 25,32% et 36,95% au prix d'augmentations significatives de la complexité de calcul de l'encodeur de 2630%(x26) et 859%(x8) par rapport au Hevc test Model (HM) 16,22 [5] dans les configurations AI et RA, respectivement. Ces gains de codage sont encore plus élevés pour les séquences vidéo HD et UHD [6].

Le processus de codage vise à résoudre le problème Rate-Distortion Optimization (RDO) qui correspond à un processus d'optimisation combinatoire. L'encodeur doit sélectionner les paramètres de codage, conduisant au coût RD minimal. Actuellement, le VTM encode un bloc de pixels avec toutes les configurations possibles d'outils de codage et retient celle qui conduit au meilleur coût RD. Ce test exhaustif entraîne des temps de codage prohibitifs. L'espace de recherche de ce problème RDO a été considérablement étendu par rapport à HEVC, notamment en raison de l'intégration de nouveaux outils de codage et de l'extension de l'espace de recherche des outils existants. Comme mentionné dans [7], le schéma de partitionnement est le plus efficace des nouveaux outils de codage VTM mais il conduit également à la plus forte augmentation de complexité.

Par rapport à HEVC, qui est basé sur un partitionnement en blocs Quad-Tree (QT), VVC intègre un schéma de partitionnement Multi-Type Tree (MTT) permettant, en plus de la division QT, des divisions BT et TT horizontales et verticales [8]. Les découpes BT et TT permettent à un bloc de pixels d'être rectangulaire au lieu d'être uniquement carré, ce qui permet au bloc de mieux s'adapter aux caractéristiques de l'image. À chaque niveau du processus de partitionnement hiérarchique, jusqu'à cinq divisions sont testées par l'encodeur, contre une seule division, c'est-à-dire la division QT, dans HEVC. Les auteurs de [9] ont montré que la désactivation des divisions BT et TT diminue le temps d'encodage

de 91,7%. Par conséquent, le processus de partitionnement offre une grande possibilité de réduction de complexité.

L'objectif de notre travail de recherche dans cette thèse était de réduire la complexité de l'encodeur VVC tout en maintenant une grande efficacité de codage. Pour réaliser un codage VVC en temps réel, plusieurs outils doivent être modifiés et notamment leurs recherches exhaustives. Un des aspects clés est la complexité du processus de partitionnement QT-MTT qui doit être réduite de manière significative. En effet, ce processus présente la plus grande opportunité de réduction de complexité parmi les outils VVC. Notre travail ouvrira la voie à la réduction de complexité de VVC par l'optimisation du processus de partitionnement. De plus, notre travail vise à donner des directives pour que les encodeur temps réel élague le partitionnement en maximisant à la fois la réduction de la complexité et la perte de qualité de l'encodage.

A.2 Contributions

Cette thèse détaille les quatre contributions qui étudient et réduisent la complexité de l'encodeur VTM. La première contribution étudie les possibilités de réduction de complexité du VTM en fonction des outils d'encodage. Les trois autres contributions cherchent à réduire la complexité de l'encodeur VTM par des techniques ML. Ces trois contributions se concentrent sur la maximisation de la réduction de complexité tout en minimisant la perte de qualité de codage de l'encodeur. Les contributions sont étudiées lors de la normalisation VVC. Elles constituent l'une des premières techniques de réduction de complexité et seront intéressantes comme base de comparaison. De plus, comme VVC apporte de nouveaux outils à haute complexité, nos techniques sont pertinentes pour les codeurs professionnels qui ont besoin de réduire fortement leur complexité. En effet, ces contributions sont destinées à inspirer les entreprises et les développeurs d'encodeurs. Les quatre contributions proposées dans cette thèse sont brièvement présentées ci-dessous.

I Les opportunités de réduction de complexité dans l'encodeur Versatile Video Coding

La réduction de la complexité est un sujet important dans le codage vidéo depuis HEVC. VVC apporte une qualité de codage impressionnante par rapport à HEVC mais au prix d'une forte augmentation de la complexité de l'encodage. En effet, lors de la normalisation de VVC, JVET a étudié de nombreux nouveaux outils de codage tels que le MTT partitionnement, 32 nouveaux modes de prédiction angulaire, et 2 nouvelles transformées. Dans ce contexte, une caractérisation et une évaluation hiérarchiques de la complexité du codage VVC avec l'encodeur VTM sont évaluées. Tout d'abord, un aperçu est présenté sur l'impact des paramètres d'encodage sur la complexité. Les paramètres étudiés comprennent la résolution spatiale, le QP, et l'impact du partitionnement. Ensuite, les possibilités de réduction de la complexité offertes par trois niveaux d'encodage sont définies et évaluées avec : le partitionnement, le mode de prédiction intra et le processus MTS sous la configuration VTM3.0 en AI. Avec le VTM10.2, le partitionnement est évalué pour les configurations AI et RA. Cette étude démontre que la complexité relative de l'encodage VTM est proportionnelle à la résolution vidéo et au QP. Elle montre également que les découpes ont un impact important sur la complexité de l'encodage VTM et la perte BD-BR.

De plus, la principale contribution a été d'étudier les limites théoriques de diverses techniques de réduction de complexité. Au début de cette thèse, cela sera utilisé pour identifier les outils intéressants sur lesquels se concentrer. Le partitionnement présente un

potentiel de réduction de complexité pouvant atteindre 97%, tandis que les pourcentages respectifs pour la prédiction intra-mode et le MTS sont de 65% et 55% avec le VTM3.0 dans la configuration AI. L'opportunité de réduction de complexité du partitionnement pour le VTM10.2 a les mêmes résultats que pour le VTM3.0 en configuration AI. En configuration RA, pour le processus de partitionnement, l'opportunité de réduction de la complexité atteint 91,2%, ce qui est proche de l'opportunité en configuration AI. De plus, sur la base de ces premiers résultats, une étude plus spécifique est proposée sur les splits et leurs profondeurs qui sera utile pour les prochaines contributions.

II Méthode de partitionnement basée sur des seuils pour les encodeurs intra Versatile Video Coding.

Cette contribution se concentre sur la réduction de la complexité de l'encodeur VTM en configuration AI. Une technique efficace basée sur un CNN est proposée pour réduire la complexité de l'encodeur VTM6.1 intra. Un CNN reçoit un bloc de luminance de 64×64 pixels et il prédit la probabilité de chaque frontière de tous les blocs de taille 4×4 pixels. A partir des probabilités aux frontières, une probabilité de découpe est déduite et comparée à un seuil pour éviter les découpes improbables. Deux solutions sont proposées pour déterminer le seuil : uniforme et non-uniforme. La solution du seuil non-uniforme fait évoluer les seuils à travers les découpe et les tailles des blocs en se basant sur les compromis souhaités entre complexité et perte de qualité. La solution proposée permet une réduction de 42,2% de la complexité pour une légère augmentation de 0,75% du BD-BR en configuration intra. Avec des séquences à haute résolution, l'accélération est encore plus grande, jusqu'à 54,5% de réduction de complexité au prix d'une perte de 0,85% de BD-BR. La solution à seuil non-uniforme atteint 54,5% de réduction de complexité pour une perte de 1,33% de BD-BR.

III Méthode de partitionnement Quad-Tree-Multi-Type Tree basée sur du machine learning pour les encodeurs intra Versatile Video Coding.

Dans cette contribution, les probabilités prédites sont traitées comme des caractéristiques spatiales. En effet, une technique efficace est proposée pour réduire la complexité du partitionnement QT-MTT. Notre approche combine un CNN à complexité modérée qui extrait les caractéristiques spatiales et des classifieurs multi-classes qui dérivent les meilleures découpes à tester dans le processus de partitionnement. Ce CNN prédit la probabilité de chaque fontière de tous les blocs de taille 4×4 pixels dans le bloc de taille 64×64. À chaque niveau du processus de partitionnement hiérarchique, un classifieur multi-classes est utilisé pour prédire, à partir de l'ensemble des probabilités de frontières, les N découpes les plus probables à explorer par l'encodeur. Un classifieur est formé pour chaque taille de bloc possible. A chaque profondeur, le nombre de découpe testée N peut être ajusté de un au nombre total de découpes possibles. Cela permet d'explorer une large gamme de compromis entre la réduction de la complexité et la perte de qualité. La solution proposée avec la configuration top-$N = 3$ atteint une réduction de la complexité de 46,6% pour une augmentation négligeable du BD-BR de 0,86%. La configuration top-$N = 2$ permet en moyenne une réduction de complexité de 69,8% pour une perte de BD-BR de 2,57%.

IV Méthode de découpe efficace basée machine learning pour les encodeurs VVC en configuration inter

Cette contribution se concentre sur la réduction de la complexité en configuration RA, ce qui modifie la sélection du partitionnement. Une méthode efficace de réduction de complexité qui repose sur des techniques de ML est proposée. Tout d'abord, un CNN prédit le partitionnement sur la base de ses $128 \times 128 \times 3$ valeurs de pixels de luminance. L'entrée est le CTU courant traité plus les deux CTUs de référence obtenus par le processus de prédiction inter. Ces deux CTUs de référence fournissent l'information temporelle utilisée pour déduire la recherche de partitionnement. Le vecteur de sortie représente le partitionnement du CTU courant. Pour une décision plus précise, des DTs sont exploités à chaque niveau de profondeur. L'information de partitionnement prédite par le CNN est donnée en entrée aux DTs en fonction de la profondeur et de la taille du bloc. Plusieurs DTs sont disponibles pour prédire les probabilités de chaque découpe, comme dans le cas d'une tâche de multi-classification. Les DTs sont capables de prédire les probabilités de découpe des blocs de taille allant de 128×128 à 4×8 ou 8×4. Enfin, en s'appuyant sur la réduction de complexité visée ou de BD-BR souhaité, un nombre de découpe non effectué est défini. Les probabilités de découpe les plus élevées sont testées à l'intérieur du VTM pour limiter la perte en BD-BR et maximiser la réduction de complexité. Notre solution $C1$ apporte 31.8% de réduction de complexité pour 1.11% de perte de BD-BR sur les Common Test Conditions (CTC).

A.3 Plan du manuscript

Cette section résume le contenu de chaque chapitre inclus dans cette thèse. Le Chapitre 2 présente la vue d'ensemble du codage vidéo qui est nécessaire pour comprendre les différentes contributions proposées dans cette thèse. Les standards de codage vidéo sont présentés et notamment le standard VVC avec ses nouveaux outils et la RDO qui optimise ses sélections au cours du processus. De plus, les CTC et les métriques permettant de comparer équitablement les résultats sont spécifiés.

Le Chapitre 3 détaille l'état de l'art des techniques qui réduisent la complexité de différentes standards. Tout d'abord, les techniques de réduction de la complexité du standard HEVC sont présentées, suivies des techniques réalisées pour Future Video Coding (FVC). FVC est un standard expérimentale de JVET, qui contient des outils de codage étudiés pour atteindre des performances de codage supérieures à celles de HEVC. Les techniques de réduction de complexité de VVC sont décrites en se concentrant sur le partitionnement, la prédiction intra-mode et le processus MTS. Une brève présentation est proposée sur les techniques qui réduisent la complexité d'autres outils. Un encodeur et un décodeur en temps réel sont également décrits. Enfin, une introduction sur les techniques basées sur le machine learning liée au codage vidéo est proposée.

Le Chapitre 4 décrit l'étude sur les opportunités de réduction de complexité. Une analyse est d'abord proposée pour déterminer l'impact de la complexité à travers plusieurs paramètres. Ensuite, les opportunités de réduction de complexité sont évaluées pour les processus de partitionnement, de prédiction intra et le MTS sous les configurations AI et RA.

Un CNN est décrit dans le Chapitre 5 qui est la base des deux chapitres suivants. La méthode et les paramètres d'apprentissage sont détaillés avec les performances du CNN. De plus, la complexité de l'inférence CNN est étudiée et optimisée afin de déterminer son impact sur le temps d'encodage. La configuration la plus rapide avec des données en

virgule flottante 16 bits atteint 16,2 Frames per Second (FPS) sur des séquences HD avec le Graphics Processing Unit (GPU) Nvidia GTX 1650 Max Q.

Une première solution de réduction de complexité est décrite dans le Chapitre 6 ainsi que ses résultats. Cette méthode se concentre sur le CNN précédent qui prédit le partitionnement grâce aux pixels de luminance du bloc considéré. Cette prédiction est représenté par un vecteur de probabilité. Une comparaison entre ces probabilités et un seuil permet de ne pas effectuer les découpes improbables. Deux approches pour la détermination du seuil sont proposées avec un seuil uniforme et un seuil non-uniforme. La première solution de réduction de complexité atteint 54,5% de réduction de complexité pour une perte de 1,33% BD-BR.

Le Chapitre 7 détaille notre méthode qui exploite les caractéristiques prédites par le CNN avec un DT pour décider de la découpe à tester. La sortie du CNN est considérée comme des caractéristiques pour optimiser la distribution de probabilité. Ces caractéristiques sont données à un DT qui prédit les probabilités de découpe. Le jeu de données utilisé est présenté avec son optimisation pour améliorer les résultats. Différentes configurations sont disponibles en fonction des N divisions effectuées dans l'encodeur. Cette méthode permet de réduire de moitié la complexité pour une perte inférieure à 1% BD-BR.

Le Chapitre 8 cherche à réduire la complexité de de l'encodage en configuration RA. Tout d'abord, un CNN est entraîné à prédire une carte de probabilité. Cette sortie est basée sur les pixels de luminance du CTU calculé plus ses deux CTUs de référence obtenus par les MVs les plus optimaux. Ensuite, la carte de probabilité est donnée à un DT qui prédit les probabilités de découpe. Comme dans la solution précédente, les N probabilités de découpe les plus élevées sont calculées pour maximiser la réduction de complexité tout en minimisant la perte BD-BR. Cette solution, en configuration RA, atteint 31,8% de réduction de complexité pour 1,11% de perte BD-BR.

Enfin, le Chapitre 9 conclut cette thèse et donne plusieurs directions pour de futures recherches.

Table des figures

1.1 Global IP traffic in exabytes per month from 2017 to 2022 by application categories. . . . 6
1.2 Internet video traffic in exabytes per month from 2017 to 2022 by resolution categories. . . . 7

2.1 Representation of YUV $4:2:0$. . . . 12
2.2 Spatial resolutions from SD (720x480) to 8K Ultra HD (7680x4320). . . . 13
2.3 Example of the same video in 60 FPS and 24 FPS for 100ms. . . . 13
2.4 Encoding and decoding process related to metrics. . . . 14
2.5 Video coding standard publication through the years. . . . 18
2.6 VVC block diagram. . . . 20
2.7 Picture divided in CTUs. . . . 22
2.8 Example of slices and tiles in a frame. . . . 23
2.9 Available split included in VTM for a $4N \times 4N$ CU. . . . 24
2.10 Example of a frame partitioning and zoom in a specific CTU. . . . 24
2.11 Wide-angle intra angular mode with non-square block. . . . 25
2.12 MV between the reference frame and the current frame. . . . 26
2.13 Application of Discrete Cosine Transform (DCT)-II on an image representing an eye. . . . 28
2.14 Application of quantization on transformed image representing an eye. . . . 29
2.15 Temporal relationship between frames on RA configuration. . . . 30
2.16 Temporal relationship between frames in Low Delay (LD) configuration. . . . 31
2.17 Rate-Distortion Optimization search process. . . . 32
2.18 Comparison between VTM and HM in BD-BR (a), encoding complexity (b), and decoding complexity (c). . . . 33

3.1 Global scheme of the state of the art for complexity reduction focusing on tree partitioning, prediction mode, and MTS. . . . 36

4.1 Average encoding time per frame in seconds as a function of the number of pixels per frame for each class and Tango2 downscaled encodings. . . . 62
4.2 Average encoding time per CTU as a function of the QP, classes A-E. . . . 63
4.3 Impact of the splits relying on the configurations defined in TABLE 4.1 on the VTM10.2. . . . 65

4.4 Impact of the splits relying on the configurations defined in TABLE 4.2 that modify the maximal split depth for MTT, BT, and TT independently on the VTM10.2. 67
4.5 Maximum complexity reduction (ΔET) of the analysed configurations over the default configuration of VTM3.0. 70

5.1 Workflow diagram of the proposed tree partitioning scheme for VVC AI coding. The input luminance block is first processed by a CNN to predict $\tilde{\mathbf{p}}$, a vector of N_p probabilities describing all edges at each 4×4 sub-block. The vector $\tilde{\mathbf{p}}$ is then processed by split decision techniques in order to determine the optimal split to process in the RDO. 76
5.2 Correspondence between the CNN output vector and a 64×64 CU partition. 77
5.3 The CNN architecture with convolution layers in orange and yellow, max-pooling layers in red, and fully connected layer in purple. 78
5.4 Representation of our dataset. (a) is a luminance 68×68 input block $\mathbf{B}$. (b) is the optimal tree partitioning of the block represented by a tree and transformed to the soft representation, i.e., a 480 elements vector $\mathbf{p}$. (c) is the hard representation, K vectors $\mathbf{c}$ of 6 probabilities, that define the optimal split for each of the K blocks in the tree. 79
5.5 Breakdown of our dataset by partition classes. (left) Unbalanced dataset. (right) Balanced dataset. 80
5.6 Decreasing loss (in X) and increasing accuracy (in Y) as a function of epoch for the training and validation set. 81
5.7 Several CNN Receiver Operating Characteristic (ROC)s for different splits and sizes on the CTC video sequences. 82
5.8 Instance of ground truth partitions and theirs corresponding CNN predictions for the sequence MarketPlace with QPs 22, 27, 32, and 37. 83

6.1 Correspondence between a predicted probability vector and a tree partitioning of a 64×64 block. Segments used to calculate the splits probabilities are also depicted in this figure. 88
6.2 Average complexity reduction and BD-BR loss computed on the CTC when modifying β of each $P(S)$ independently. 89
6.3 Complexity reduction and BD-BR loss of each CTC class when modifying β of each $P(S)$ independently. 91
6.4 Average complexity reduction and BD-BR loss computed on the CTC when modifying the threshold of $P(QT)$ for each size independently. 92
6.5 Average complexity reduction and BD-BR loss computed on the CTC when modifying the threshold of $P(BT)$ for each size independently. 92
6.6 Average complexity reduction and BD-BR loss computed on the CTC when modifying the threshold of $P(TT)$ for each size independently. 93
6.7 Local search algorithm based on gradient descent to define the threshold β. 94
6.8 Performance comparison between our proposed uniform solution, non-uniform solution, and state of the art techniques. Circles are averaged results of Full Hd and 4K sequences. Crosses are averaged results of CTC classes without class F. 97

7.1 Workflow diagram of the proposed tree partitioning scheme for VVC AI coding. The input luminance block is first processed by a CNN to predict $\tilde{\mathbf{p}}$, a vector of N_p probabilities describing all edges at each 4×4 sub-block. The vector $\tilde{\mathbf{p}}$ is then processed by a decision tree LightGBM to predict probabilities of the six partitioning options through the vector $\tilde{\mathbf{c}}$. The top-N splits with the highest probabilities are tested by the RD process of the encoder to select the optimal split in terms of RD-cost. 100
7.2 Complexity reduction versus BD-BR performance comparison between the proposed method and the state of the art techniques averaged on the CTC classes without class F (AI configuration). An interpolation curve over our four configurations is plotted in blue dot line. 106

8.1 Workflow diagram of the proposed fast tree partitioning scheme for VVC under RA configuration. The current luminance CTU B_{cur} plus its two CTUs of reference B_{MV1} and B_{MV2} which are processed by a CNN to predict $\tilde{\mathbf{p}}$, a vector of N_p probabilities describing all edges at each 4×4 block. The vector $\tilde{\mathbf{p}}$ is then processed by the Multi-Class Classifier (MCC) to predict probabilities of the six partitioning options through the vector $\tilde{\mathbf{c}}$. The top-N splits with the highest probabilities are tested by the RD optimization of the encoder to select the optimal split in terms of RD-cost. 110
8.2 Representation of our dataset. (a) is the input block $\mathbf{B}$. (b) is the optimal tree partitioning of the block represented by a tree and transformed to the soft representation, i.e., a 1984 elements vector $\mathbf{p}$. (c) is the hard representation, K vectors $\mathbf{c}$ of 6 probabilities, that define the optimal split for each of the K blocks in the tree. K is the decision tree depth. 114
8.3 Breakdown of our inter prediction dataset by partition classes. (left) Unbalanced dataset. (right) Balanced dataset. 115
8.4 The process of data augmentation for the CNN training. In top-left the reference tree partitioning. In top-right the horizontal flip of the reference. In bottom-left the vertical flip of the reference. In bottom-right the horizontal plus vertical flip of the reference. 116
8.5 Several losses from the different CNNs trained with different parameters. Top-left, the MobileNetV2 loss with the weight pre-trained on imagenet. Top-right, the MobileNetV2 loss without pre-training. Bottom-left, the technicolor CNN loss. Bottom-right, the EfficientNet loss. 118
8.6 Performance comparison between the proposed solution through different configurations and the state of the art techniques. Circles are average results of Full Hd and 4K sequences. Crosses are an average through the CTC classes without class F. 123

Liste des tableaux

2.1 CTC sequences with their respective width, height, and bit depth. 17

3.1 Main features and performance of state of the art complexity reduction techniques for HM software under AI and RA configurations. 37
3.2 Main features and performance of state of the art complexity reduction techniques under Joint Exploration Model (JEM) software. 41
3.3 Main features and performances of state of the art complexity reduction techniques under VTM software. 47

4.1 Description of the five configurations proposed to show the impact of the split in the tree partitioning process relying of the maximal depth allowed. . 64
4.2 Description of the seven configurations proposed to show the impact of the split depth in the tree partitioning process relying of the maximal depth allowed. 66
4.3 Analysed configurations with maximum available complexity reduction. . . . 69
4.4 Complexity reduction opportunities for the tree partitioning process with the VTM10.2 under AI and RA configurations through the CTC. 72

5.1 Breakdown of our dataset by resolution. 78
5.2 Inference time under the tensorflow framework depending on the Central Processing Unit (CPU) exploited. 84
5.3 Inference time under the frugally-deep framework depending on the activated compilation flag. 84
5.4 Inference time with TensorRT depending on the precision selected. 85

6.1 Performance of the proposed solution in VTM6.1 with different uniform threshold values β in comparison with state of the art techniques. 95
6.2 Description of the thresholds for the configuration C1 and C2 under the several splits and sizes available. 96

7.1 Possible splits according to the CU size in addition to the no-split option . . 101
7.2 Performance of every DT LightGBM (LBGM) models with their defined size and number of output through top-1, top-2, and top-3 accuracy on the validation set. 103

7.3 ΔET and BD-BR performance of the proposed solution in comparison with state of the art techniques in AI coding configuration. 104
7.4 Overheads of the CNN and DT LBGM (in %) for C_2 configuration with respect to the execution time of the VTM anchor. 107

8.1 Possible splits according to the CU size 112
8.2 Sequences encoded to create our dataset selected from Xiph [110] and UVG [111] datasets. 113
8.3 Performance of every MCCs with their defined sizes and number of outputs through top-1, top-2, and top-3 accuracies on the validation set. 120
8.4 The configurations performed in the VTM depending on the number of split performed. 120
8.5 ΔET and BD-BR performance of the proposed solution in comparison with state of the art techniques in RA coding configuration. 122

Bibliographie

[1] Cisco, "Cisco visual networking index : Forecast and trends, 2017-2022," . 5, 133

[2] Benjamin Bross, Ye-Kui Wang, Yan Ye, Shan Liu, Jianle Chen, Gary J. Sullivan, and Jens-Rainer Ohm, "Overview of the versatile video coding (VVC) standard and its applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 10, pp. 3736–3764. 6, 134

[3] D. García-Lucas, G. Cebrián-Márquez, and P. Cuenca, "Rate-distortion/complexity analysis of HEVC, VVC and AV1 video codecs," *Multimedia Tools and Applications*. 6, 134

[4] F. Zhang, A. Katsenou, M. Afonso, G. Dimitrov, and D. Bull, "Comparing VVC, HEVC and AV1 using objective and subjective assessments," *arXiv :2003.10282 [eess]*. 6, 134

[5] F. Bossen, X. Li, and K. Suehring, "AHG report : Test model software development (AHG3)," *JVET-T0003*. 6, 134

[6] Naty Sidaty, Wassim Hamidouche, Olivier Deforges, Pierrick Philippe, and Jerome Fournier, "Compression performance of the versatile video coding : HD and UHD visual quality monitoring," in *2019 Picture Coding Symposium (PCS)*. pp. 1–5, IEEE. 6, 134

[7] E. François, M. Kerdranvat, R. Jullian, and C. Chevance, "VVC PER-TOOL PERFORMANCE EVALUATION COMPARED TO HEVC," p. 14. 7, 46, 134

[8] A. Wieckowski, J. Ma, H. Schwarz, D. Marpe, and T. Wiegand, "Fast partitioning decision strategies for the upcoming versatile video coding (VVC) standard," in *2019 IEEE International Conference on Image Processing (ICIP)*, pp. 4130–4134. 7, 23, 79, 104, 134

[9] M. Saldanha, G. Sanchez, C. Marcon, and L. Agostini, "Complexity analysis of VVC intra coding," in *2020 IEEE International Conference on Image Processing (ICIP)*. pp. 3119–3123, IEEE. 7, 79, 134

[10] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves," *VCEG-M33*. 13, 102, 117

[11] Z. Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli, "Image quality assessment : From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612. 15

[12] Zhi Li, Christos Bampis, Julie Novak, Anne Aaron, Kyle Swanson, Anush Moorthy, and Jan De Cock, "VMAF : The journey continues," p. 12. 15

[13] K. Seshadrinathan and A.C. Bovik, "Motion tuned spatio-temporal quality assessment of natural videos," *IEEE Transactions on Image Processing*, vol. 19, no. 2, pp. 335–350. 15

[14] Michail-Alexandros Kourtis, Harilaos Koumaras, and Fidel Liberal, "Reduced-reference video quality assessment using a static video pattern," *Journal of Electronic Imaging*, vol. 25, no. 4, pp. 043011. 15

[15] A. Mittal, R. Soundararajan, and A. C. Bovik, "Making a "completely blind" image quality analyzer," *IEEE Signal Processing Letters*, vol. 20, no. 3, pp. 209–212. 15

[16] Michele A. Saad, Alan C. Bovik, and Christophe Charrier, "Blind prediction of natural video quality," *IEEE Transactions on Image Processing*, vol. 23, no. 3, pp. 1352–1365. 15

[17] ITU-T, "Methods for the subjective assessment of video quality, audio quality and audiovisual quality of internet video and distribution quality television in any environment," . 16

[18] ITU-R, "RECOMMENDATION ITU-r BT.500-14(- methodologies for the subjective assessment of the quality of television images," p. 102. 16

[19] F. Bossen, J. Boyce, K. Suehring, X. Li, and V. Seregin, "JVET common test conditions and software reference configurations for SDR video," *JVET-M1010*. 16, 62, 102

[20] Bitmovin, "Bitmovin video developer report 2019," . 19

[21] Michel Kerdranvat, "THE VIDEO CODEC LANDSCAPE IN 2020," vol. 3, pp. 11. 19

[22] M. Wang, J. Li, L. Zhang, K. Zhang, H. Liu, S. Wang, S. Kwong, and S. Ma, "Extended coding unit partitioning for future video coding," *IEEE Transactions on Image Processing*, vol. 29, pp. 2931–2946. 22

[23] Moonmo Koo, Mehdi Salehifar, Jaehyun Lim, and Seung-Hwan Kim, "Low frequency non-separable transform (LFNST)," in *2019 Picture Coding Symposium (PCS)*. pp. 1–5, IEEE. 28, 95

[24] Andrey Norkin, Gisle Bjontegaard, Arild Fuldseth, Matthias Narroschke, Masaru Ikeda, Kenneth Andersson, Minhua Zhou, and Geert Van der Auwera, "HEVC deblocking filter," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1746–1754. 29

[25] Chih-Ming Fu, Ching-Yeh Chen, Yu-Wen Huang, and Shawmin Lei, "Sample adaptive offset for HEVC," in *2011 IEEE 13th International Workshop on Multimedia Signal Processing*. pp. 1–5, IEEE. 29

[26] Chia-Yang Tsai, Ching-Yeh Chen, Tomoo Yamakage, In Suk Chong, Yu-Wen Huang, Chih-Ming Fu, Takayuki Itoh, Takashi Watanabe, Takeshi Chujoh, Marta Karczewicz, and Shaw-Min Lei, "Adaptive loop filtering for video coding," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 6, pp. 934–945. 29

[27] Taoran Lu, Fangjun Pu, Peng Yin, Sean McCarthy, Walt Husak, Tao Chen, Edouard Francois, Christophe Chevance, Franck Hiron, Jie Chen, Ru-Ling Liao, Yan Ye, and Jiancong Luo, "Luma mapping with chroma scaling in versatile video coding," in *2020 Data Compression Conference (DCC)*. pp. 193–202, IEEE. 29

[28] A. Mercat, F. Arrestier, W. Hamidouche, M. Pelcat, and D. Menard, "Energy reduction opportunities in an HEVC real-time encoder," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1158–1162. 36

[29] Biao Min and R. Cheung, "A fast CU size decision algorithm for the HEVC intra encoder," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 5, pp. 892–896. 37

[30] Yan Huang, Li Song, and Ebroul Izquierdo, "CNN accelerated intra video coding, where is the upper bound ?," p. 5. 37, 39

[31] Aolin Feng, Changsheng Gao, Li Li, Dong Liu, and Feng Wu, "Cnn-based depth map prediction for fast block partitioning in HEVC intra coding," in *2021 IEEE International Conference on Multimedia and Expo (ICME)*. pp. 1–6, IEEE. 37, 39

[32] M. Xu, T. Li, Z. Wang, X. Deng, R. Yang, and Z. Guan, "Reducing complexity of HEVC : A deep learning approach," *IEEE Transactions on Image Processing*, vol. 27, no. 10, pp. 5044–5059. 37, 39

[33] B. Huang, Z. Chen, Q. Cai, M. Zheng, and D. Wu, "Rate-distortion-complexity optimized coding mode decision for HEVC," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1. 37, 71

[34] G. Correa, P. Assuncao, L. Agostini, and L. da Silva Cruz, "Fast hevc encoding decisions using data mining," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 4, pp. 660–673, 2015. 37, 38, 71

[35] Guilherme Correa, Pargles Dall'Oglio, Daniel Palomino, and Luciano Agostini, "Fast block size decision for HEVC encoders with on-the-fly trained classifiers," in *2020 28th European Signal Processing Conference (EUSIPCO)*. pp. 540–544, IEEE. 36, 37, 38

[36] Mateus Grellert, Bruno Zatt, Sergio Bampi, and Luis A. da Silva Cruz, "Fast coding unit partition decision for HEVC using support vector machines," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 6, pp. 1741–1753. 37, 38

[37] H. Zhang and Z. Ma, "Fast intra mode decision for high efficiency video coding (HEVC)," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 4, pp. 660–668. 37, 40

[38] S. Ryu and J. Kang, "Machine learning-based fast angular prediction mode decision technique in video coding," *IEEE Transactions on Image Processing*, vol. 27, no. 11, pp. 5525–5538. 37, 40

[39] Nan Song, Zhenyu Liu, Xiangyang Ji, and Dongsheng Wang, "CNN oriented fast PU mode decision for HEVC hardwired intra encoder," in *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. pp. 239–243, IEEE. 37, 40

[40] Jaehwan Kim, Jungyoup Yang, Kwanghyun Won, and Byeungwoo Jeon, "Early determination of mode decision for HEVC," in *2012 Picture Coding Symposium*. pp. 449–452, IEEE. 37, 40

[41] T. Li, M. Xu, X. Deng, and L. Shen, "Accelerate CTU partition to real time for HEVC encoding with complexity control," *IEEE Transactions on Image Processing*, vol. 29, pp. 7482–7496. 39

[42] T. Lin, H. Jiang, J. Huang, and P. Chang, "Fast binary tree partition decision in h.266/FVC intra coding," in *2018 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, pp. 1–2. 41

[43] J. Chen, Y. Chiu, C. Lee, and Y. Tsai, "Utilize neighboring LCU depth information to speedup FVC/h.266 intra coding," in *2019 International Conference on System Science and Engineering (ICSSE)*, pp. 308–312. 41, 42

[44] Zongju Peng, Chao Huang, Fen Chen, Gangyi Jiang, Xin Cui, and Mei Yu, "Multiple classifier-based fast coding unit partition for intra coding in future video coding," *Signal Processing : Image Communication*, vol. 78, pp. 171–179. 41, 43

[45] Maraoui Amna, Werda Imen, Sayadi Fatma Ezahra, and Atri Mohamed, "Fast intra-coding unit partition decision in h.266/FVC based on deep learning," *Journal of Real-Time Image Processing*. 41, 44

[46] Z. Jin, P. An, L. Shen, and C. Yang, "CNN oriented fast QTBT partition algorithm for JVET intra coding," in *2017 IEEE Visual Communications and Image Processing (VCIP)*, pp. 1–4. 41, 44

[47] Z. Wang, S. Wang, J. Zhang, S. Wang, and S. Ma, "Probabilistic decision based block partitioning for future video coding," *IEEE Transactions on Image Processing*, vol. 27, no. 3, pp. 1475–1486. 41, 42

[48] Thomas Amestoy, Alexandre Mercat, Wassim Hamidouche, Cyril Bergeron, and Daniel Menard, "Random forest oriented fast QTBT frame partitioning," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. pp. 1837–1841, IEEE. 41, 42

[49] Z. Wang, S. Wang, J. Zhang, S. Wang, and S. Ma, "Effective quadtree plus binary tree block partition decision for future video coding," in *2017 Data Compression Conference (DCC)*, pp. 23–32. 41, 43

[50] Z. Wang, S. Wang, X. Zhang, S. Wang, and S. Ma, "Fast QTBT partitioning decision for interframe coding with convolution neural network," in *2018 25th IEEE International Conference on Image Processing (ICIP)*, pp. 2550–2554. 41, 44

[51] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv :1512.03385 [cs]*. 44, 77, 117

[52] F. Galpin, F. Racapé, S. Jaiswal, P. Bordes, F. Le Léannec, and E. François, "CNN-based driving of block partitioning for intra slices encoding," in *2019 Data Compression Conference (DCC)*, pp. 162–171. 45

[53] Frank Bossen, Karsten Suhring, Adam Wieckowski, and Shan Liu, "VVC complexity and software implementation analysis," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1. 45

[54] Junan Chen, Heming Sun, Jiro Katto, Xiaoyang Zeng, and Yibo Fan, "Fast QTMT partition decision algorithm in VVC intra coding based on variance and gradient," in *2019 IEEE Visual Communications and Image Processing (VCIP)*. pp. 1–4, IEEE. 47

[55] Jing Cui, Tao Zhang, Chenchen Gu, Xinfeng Zhang, and Siwei Ma, "Gradient-based early termination of CU partition in VVC intra coding," in *2020 Data Compression Conference (DCC)*. pp. 103–112, IEEE. 47, 48

[56] T. Fu, H. Zhang, F. Mu, and H. Chen, "Fast CU partitioning algorithm for h.266/VVC intra-frame coding," in *2019 IEEE International Conference on Multimedia and Expo (ICME)*. pp. 55–60, IEEE. 47, 48, 102, 104, 105, 107

[57] F. Chen, Y. Ren, Z. Peng, G. Jiang, and X. Cui, "A fast CU size decision algorithm for VVC intra prediction based on support vector machine," *Multimedia Tools and Applications*. 47, 49, 102, 104, 105, 107

[58] H. Yang, L. Shen, X. Dong, Q. Ding, P. An, and G. Jiang, "Low-complexity ctu partition structure decision and fast intra mode decision for versatile video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 6, pp. 1668–1682, 2020. 47, 49, 51, 76, 102, 107

[59] Genwei Tang, Minge Jing, Xiaoyang Zeng, and Yibo Fan, "Adaptive CU split decision with pooling-variable CNN for VVC intra encoding," in *2019 IEEE Visual Communications and Image Processing (VCIP)*. pp. 1–4, IEEE. 47, 50

[60] J. Zhao, Y. Wang, and Q. Zhang, "Adaptive CU split decision based on deep learning and multifeature fusion for h.266/VVC," *Scientific Programming*, vol. 2020, pp. 1–11. 47, 50

[61] T. Li, M. Xu, R. Tang, Y. Chen, and Q. Xing, "Deepqtmt : A deep learning approach for fast qtmt-based cu partition of intra-mode vvc," *IEEE Transactions on Image Processing*, vol. 30, pp. 5377–5390, 2021. 47, 50, 102, 104, 105, 107

[62] Na Tang, Jian Cao, Fan Liang, Jun Wang, Hongmei Liu, Xiaoyang Wang, and Xiaorong Du, "Fast CTU partition decision algorithm for VVC intra and inter coding," in *2019 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*. pp. 361–364, IEEE. 47, 119, 121, 122, 123

[63] T. Amestoy, A. Mercat, W. Hamidouche, D. Menard, and C. Bergeron, "Tunable VVC frame partitioning based on lightweight machine learning," *IEEE Transactions on Image Processing*, vol. 29, pp. 1313–1328. 46, 47, 48, 76

[64] Zhaoqing Pan, Peihan Zhang, Bo Peng, Nam Ling, and Jianjun Lei, "A CNN-based fast inter coding method for VVC," *IEEE Signal Processing Letters*, pp. 1–1. 47, 49, 119, 121, 122, 123

[65] Q. Zhang, Y. Wang, L. Huang, and B. Jiang, "Fast CU partition and intra mode decision method for h.266/VVC," *IEEE Access*, vol. 8, pp. 117539–117550. 47, 51

[66] T. Fu, H. Zhang, F. Mu, and H. Chen, "Two-stage fast multiple transform selection algorithm for VVC intra coding," in *2019 IEEE International Conference on Multimedia and Expo (ICME)*. pp. 61–66, IEEE. 47, 51

[67] E. Agustsson and R. Timofte, "NTIRE 2017 challenge on single image super-resolution : Dataset and study," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. pp. 1122–1131, IEEE. 47, 78

[68] W. Hamidouche, P. Philippe, C.-E. Mohamed, A. Kammoun, D. Menard, and O. Deforges, "Hardware-friendly DST-VII/DCT-VIII approximations for the versatile video coding standard," in *2019 Picture Coding Symposium (PCS)*. pp. 1–5, IEEE. 52

[69] Meng Wang, Shiqi Wang, Junru Li, Li Zhang, Yue Wang, Siwei Ma, and Sam Kwong, "Low complexity trellis-coded quantization in versatile video coding," *IEEE Transactions on Image Processing*, vol. 30, pp. 2378–2393. 52

[70] Thomas Amestoy, Wassim Hamidouche, Cyril Bergeron, and Daniel Menard, "Quality-driven dynamic VVC frame partitioning for efficient parallel processing," in *2020 IEEE International Conference on Image Processing (ICIP)*. pp. 3129–3133, IEEE. 52

[71] Jens Brandenburg, Adam Wieckowski, Tobias Hinz, Anastasia Henkel, Valeri George, Ivan Zupancic, Christian Stoffers, Benjamin Bross, Heiko Schwarz, and Detlev Marpe, "Towards fast and efficient VVC encoding," in *2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP)*. pp. 1–6, IEEE. 53

[72] Adam Wieckowski, Jens Brandenburg, Tobias Hinz, Christian Bartnik, Valeri George, Gabriel Hege, Christian Helmrich, Anastasia Henkel, Christian Lehmann, Christian Stoffers, Ivan Zupancic, Benjamin Bross, and Detlev Marpe, "Vvenc : An open and optimized vvc encoder implementation," in *2021 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*. pp. 1–2, IEEE. 53

[73] Christian R Helmrich, Sebastian Bosse, Heiko Schwarz, Detlev Marpe, and Thomas Wiegand, "A STUDY OF THE EXTENDED PERCEPTUALLY WEIGHTED PEAK SIGNAL-TO-NOISE RATIO (XPSNR) FOR VIDEO COMPRESSION WITH DIFFERENT RESOLUTIONS AND BIT-DEPTHS," p. 9. 53

[74] Bin Zhu, Shan Liu, Yuan Liu, Yi Luo, Jing Ye, Haiyan Xu, Ying Huang, Hualong Jiao, Xiaozhong Xu, Xianguo Zhang, and Chenchen Gu, "A real-time h.266/VVC software decoder," in *2021 IEEE International Conference on Multimedia and Expo (ICME)*. pp. 1–6, IEEE. 54

[75] Guo Lu, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Chunlei Cai, and Zhiyong Gao, "DVC : An end-to-end deep video compression framework," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 10998–11007, IEEE. 54

[76] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli, "End-to-end optimized image compression," *arXiv :1611.01704 [cs, math]*. 54, 55

[77] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston, "Variational image compression with a scale hyperprior," *arXiv :1802.01436 [cs, eess, math]*. 54

[78] Oren Rippel, Sanjay Nair, Carissa Lew, Steve Branson, Alexander Anderson, and Lubomir Bourdev, "Learned video compression," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. pp. 3453–3462, IEEE. 54

[79] Eirikur Agustsson, David Minnen, Nick Johnston, Johannes Balle, Sung Jin Hwang, and George Toderici, "Scale-space flow for end-to-end optimized video compression," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 8500–8509, IEEE. 55

[80] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár, "Lossy image compression with compressive autoencoders," *arXiv :1703.00395 [cs, stat]*. 55

[81] Jean Bégaint, Fabien Racapé, Simon Feltman, and Akshay Pushparaja, "CompressAI : a PyTorch library and evaluation platform for end-to-end compression research," *arXiv :2011.03029 [cs, eess]*. 55

[82] Jiahao Li, Bin Li, Jizheng Xu, Ruiqin Xiong, and Wen Gao, "Fully connected network-based intra prediction for image coding," *IEEE Transactions on Image Processing*, vol. 27, no. 7, pp. 3236–3247. 55

[83] Maria Santamaria, Saverio Blasi, Ebroul Izquierdo, and Marta Mrak, "Analytic simplification of neural network based intra-prediction modes for video compression," in *2020 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*. pp. 1–4, IEEE. 56

[84] Yang Wang, Xiaopeng Fan, Chuanmin Jia, Debin Zhao, and Wen Gao, "Neural network based inter prediction for HEVC," in *2018 IEEE International Conference on Multimedia and Expo (ICME)*. pp. 1–6, IEEE. 56

[85] Martin Benjak, Holger Meuel, Thorsten Laude, and Jorn Ostermann, "Enhanced machine learning-based inter coding for VVC," in *2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*. pp. 021–025, IEEE. 56

[86] William Lotter, Gabriel Kreiman, and David Cox, "Deep predictive coding networks for video prediction and unsupervised learning," *arXiv :1605.08104 [cs, q-bio]*. 56

[87] A Geiger, P Lenz, C Stiller, and R Urtasun, "Vision meets robotics : The KITTI dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237. 56

[88] Soulef Bouaafia, Seifeddine Messaoud, Randa Khemiri, and Fatma Elzahra Sayadi, "VVC in-loop filtering based on deep convolutional neural network," *Computational Intelligence and Neuroscience*, vol. 2021, pp. 1–9. 56

[89] Fan Zhang, Di Ma, Chen Feng, and David R. Bull, "Video compression with CNN-based post processing," *IEEE MultiMedia*, pp. 1–1. 57

[90] Man M. Ho, Jinjia Zhou, and Gang He, "RR-DnCNN v2.0 : Enhanced restoration-reconstruction deep neural network for down-sampling-based video coding," *IEEE Transactions on Image Processing*, vol. 30, pp. 1702–1715. 57

[91] F. Racapé, G. Rath, F. Urban, L. Zhao, S. Liu, X. Zhao, x. Li, A. Filippov, v. Rufitskiy, and J. Chen, "CE3-related wide-angle intra prediction for non-square blocks," *JVET-K0500*. 61

[92] W. Chien, J. Boyce, R. Chernyak, K. Francois, R. Hashimoto, Y. He, Y. Huang, and S. Liu, "JVET AHG report : Tool reporting procedure," *JVET-L0013*. 61

[93] B. Bross, P. Keydel, H. Schwarz, D. Marpe, T. Wiegand, L. Zhao, X. Zhao, X. Li, S. Liu, Y. Chang, H. Jiang, P. Lin, C. Kuo, C. Lin, and C. Lin, "Multiple reference line intra prediction," *JVET-L0283*. 77

[94] Mingxing Tan and Quoc V. Le, "EfficientNet rethinking model scaling for convolutional neural networks," *arXiv :1905.11946 [cs, stat]*. 78, 117

[95] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv :1409.1556 [cs]*. 78

[96] E. Makov, *Dataset image 4k*. 79

[97] Iec Jtc and Itu-T Sg, "Call for evidence on learning-based image coding technologies (JPEG AI)," p. 15. 79

[98] S. Hasinoff, D. Sharlet, R. Geiss, A. Adams, J. Barron, F. Kainz, J. Chen, and M. Levoy, "Burst photography for high dynamic range and low-light imaging on mobile cameras," *ACM Transactions on Graphics*, vol. 35, no. 6, pp. 1–12. 79

[99] R. Timofte, L. Gool, M. Yang, L. Zhang, B. Lim, S. Son, H. Kim, S. Nah, K. Lee, X. Wang, Y. Tian, K. Yu, Y. Zhang, S. Wu, C. Dong, L. Lin, Y. Qiao, C. Loy, W. Bae, J. Yoo, Y. Han, J. Ye, J. Choi, M. Kim, Y. Fan, J. Yu, W. Han, D. Liu, H. Yu, Z. Wang, H. Shi, X. Wang, T. Huang, Y. Chen, K. Zhang, W. Zuo, Z. Tang, L. Luo, S. Li, M. Fu, L. Cao, W. Heng, G. Bui, T. Le, Y. Duan, D. Tao, R. Wang, X. Lin, J. Pang, J. Xu, Y. Zhao, X. Xu, J. Pan, D. Sun, Y. Zhang, X. Song, Y. Dai, X. Qin, X. Huynh, T. Guo, H. Mousavi, T. Vu, V. Monga, C. Cruz, K. Egiazarian, V. Katkovnik, R. Mehta, A. Jain, A. Agarwalla, C. Praveen, R. Zhou, H. Wen, C. Zhu, Z. Xia, Z. Wang, and Q. Guo, "NTIRE 2017 challenge on single image super-resolution : Methods and results," p. 12. 79

[100] F. Chollet et al., *Keras*. 80, 116

[101] F. Abadi et al., "TensorFlow : Large-scale machine learning on heterogeneous distributed systems," p. 19. 80, 116

[102] D. Kingma and J. Ba, "Adam : A method for stochastic optimization," *arXiv :1412.6980 [cs]*. 81, 116

[103] T. Hermann, *Frugally Deep*. 95, 102, 117

[104] M. Lei, F. Luo, X. Zhang, S. Wang, and S. Ma, "Look-ahead prediction based coding unit size pruning for VVC intra coding," in *2019 IEEE International Conference on Image Processing (ICIP)*, pp. 4120–4124. 95, 96, 98

[105] Sang-Hyo Park and Je-Won Kang, "Context-based ternary tree decision method in versatile video coding for fast intra coding," *IEEE Access*, vol. 7, pp. 172597–172605. 95, 96, 98

[106] Santiago De-Luxan-Hernandez, Valeri George, Jackie Ma, Tung Nguyen, Heiko Schwarz, Detlev Marpe, and Thomas Wiegand, "Intra sub-partitions coding mode," *JVET-M0102*. 95

[107] Christian Helmrich, Heiko Schwarz, Tung Nguyen, Christian Rudat, Detlev Marpe, Thomas Wiegand, Bappaditya Ray, Geert Van der Auwera, Adarsh Ramasubramonian, Muhammed Coban, and Marta Karczewicz, "Joint chroma residual coding with multiple modes," *JVET-O0105*. 95

[108] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T. Liu, "LightGBM : A highly efficient gradient boosting decision tree," *Advances in Neural Information Processing Systems 30*, p. 9. 101, 102, 116, 117

[109] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen, "MobileNetV2 : Inverted residuals and linear bottlenecks," *arXiv :1801.04381 [cs]*. 112, 117

[110] Chris Montgomery and others, "Xiph. org video test media (derf's collection), the xiph open source community, 1994," *Online, https ://media. xiph. org/video/derf*, vol. 3. 113, 144

[111] Alexandre Mercat, Marko Viitanen, and Jarno Vanne, "UVG dataset : 50/120fps 4k sequences for video codec analysis and development," in *Proceedings of the 11th ACM Multimedia Systems Conference*. pp. 297–302, ACM. 113, 144

DOCTORAT BRETAGNE LOIRE / MATHSTIC

INSA RENNES

Titre : Réduction de complexité de l'encodage vidéo VVC à l'aide de techniques d'apprentissage automatique

Mots-clés : Codage vidéo, VVC, Réduction de la complexité, Apprentissage machine
Résumé : La visualisation de contenu vidéo a été révolutionnée en une décennie avec l'émergence de nouveaux services tels que la vidéo à la demande, le streaming vidéo ou les plateformes de partage vidéo. Les nouveaux formats vidéo et l'explosion du trafic vidéo IP nécessitent de nouvelles techniques de compression vidéo encore plus efficaces que les techniques existantes. L'organisation JVET ISO/IEC, ITU-T a standardisé en juillet 2020 la norme de codage vidéo VVC ITU-T H.266 comme successeur de HEVC. Les nouveaux outils inclus dans VVC permettent une réduction de près de 40% du débit mais au détriment d'une augmentation significative de la complexité de calcul de l'encodeur, estimée à 859% (x8) par rapport à HEVC.

L'objectif de ce travail de recherche est de proposer des techniques de réduction de la complexité de l'encodage VVC tout en minimisant la perte de qualité du codage. La première contribution analyse les possibilités de réduction de complexité du processus d'encodage VVC en fonction des outils de codage. Les contributions suivantes se concentre sur la réduction de la complexité du processus de partitionnement en arbre QT-MTT à l'aide de techniques de ML. Tout d'abord, une solution basée sur un CNN est développée afin d'extraire les probabilités de partitionnement associées aux différents segments présents au sein d'un CTU dans le cas d'une configuration AI. Une première technique de réduction de complexité exploitant ce CNN et utilisant un ensemble de seuils pré-calculés pour les prises de décision de partitionnement a été définie. Une seconde technique utilisant une approche de ML basée sur des arbres de décision (DT) pour la prise de décision de partitionnement est proposée. Finalement, une technique intégrant un CNN et une approche de ML basée DT est proposée dans le cas d'une configuration inter.

Title: Complexity reduction of VVC video encoding using machine learning techniques

Keywords: Video coding, VVC, Complexity reduction, Machine learning
Abstract: The video content visualization has been revolutionized in a decade with the emergence of new services like video on demand, video streaming, or video sharing platforms. The emerging video formats alongside with the explosion of IP video traffic require new video coding techniques that are even more efficient than existing ones. The organization JVET ISO/IEC, ITU-T standardized in July 2020 the video coding standard VVC ITU-T H.266 as a successor to HEVC. The new tools included in VVC provides almost 40% bitrate reduction but at the expense of a significant increase in encoder computational complexity estimated to 859%(x8) compared with HEVC.

The research work objective is to propose complexity reduction techniques for VVC encoding while minimizing the coding quality degradation. The first contribution analyzes the possibilities of complexity reduction for the VVC encoding process depending on the encoding tools. The next contributions focus on reducing the complexity of the QT-MTT

partitioning process using ML techniques. First, a CNN-based solution is developed to extract the partitioning probabilities associated with the different segments located within a CTU in the case of an AI configuration. A first complexity reduction technique exploiting this CNN and using a set of pre-computed thresholds for partitioning decisions has been defined. A second technique using a DT based ML approach for partitioning decision making is proposed. Finally, a technique integrating a CNN and a DT-based ML approach is proposed in the case of an inter configuration.