A MACHINE LEARNING APPROACH FOR FAST CU PARTITIONING AND TIME COMPLEXITY REDUCTION IN VIDEO CODING

Md. Zahirul Islam[#], Md. Eimran Hossain Eimon[#], Boshir Ahmed[#]

[#]Rajshahi University of Engineering & Technology, Rajshahi, Bangladesh

ABSTRACT

Modern video codecs such as HEVC have high capability to minimize the bits and also have an immense complexity because of testing more combinations during the RDO (Rate Distortion Optimization) process. Partitioning a coding unit (CU) into smaller ones is the leading cause of the increased time complexity. For this reason, we need a fast and efficient algorithm for real-time applications. In this paper, we proposed an approach for reducing the time complexity of CU splitting by using decision tree classifier. By using the proposed method, it is possible to skip a block if the block contains only a single motion, i.e. if the block falls in a homogeneous region. Experimental results show that on average 45.02% time complexity reduction is possible by implementing the skip criterion. However, as a trade-off on average 1.38% bit rate is increased over standalone HEVC.

Index Terms— Video Coding, Complexity Reduction, Motion Discontinuity, Machine Learning, Decision Tree.

1. INTRODUCTION

A natural video has multiple objects and each object has different shapes or boundaries, depths, illumination and texture. Typical video scene has both spatial and temporal characteristics. Spatial category includes the number of objects and shapes, color and texture variation in the scene; whereas the temporal category includes camera motion, objects movement and illumination change. If a coding unit contains multiple motions such as camera motion, movement of the objects or object boundaries then further partitioning is occurred in the traditional video codecs during the RDO process. Containing multiple motions in a block is defined as Motion Discontinuity in this paper. On the other hand, if a CU contains only a single motion, i.e. the CU falls in a uniform or homogeneous region then there is no need to split this CU again.

Modern video coding standards such as HEVC [1] and AV1 [2] have high efficiency of bit reduction and increased time complexity compared to its previous standards or predecessors like AVC. The high efficiency of modern codecs, including VVC is achieved by having the capability of flexible partitioning so-called coding quad-tree [3]. For example, HEVC partitions the CUs, which are containing multiple motions for the reduction of bits. This concept can be said in another way: a region with relative motions or object boundaries needs more bits for describing it and needs partitioning. The opposite is true for the region with no motions or no object boundaries, i.e. containing single motion (uniform region), and this paradigm is illustrated in fig. 1. A CU partitioning scheme is shown in fig. 2, where partitioning occurs in-depth 0, 1 and 2 that's are represented by mid-gray marked block respectively [1]. To decide a CU is split or not, the cost function of rate-distortion optimization (RDO) is examined for all possible combinations and this optimization loop is continued

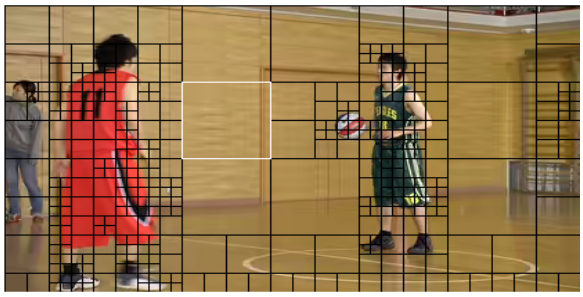

Fig. 1. An example of CTU (Coding Tree Unit) partitioning scheme by the permission of elecard software. White boundary block is an example of homogeneous region.

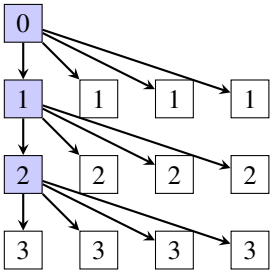

Fig. 2. A quad-tree partitioning structure with corresponding depth. 0 marked block size is 64 × 64 and corresponding depth is 0 (root node), all 1 marked block size is 32 × 32 and corresponding depth is 1, all 2 marked block size is 16 × 16 and corresponding depth is 2, all 3 marked block size is 8 × 8 and corresponding depth is 3 (leaf node).

until the optimal values are found that optimize J for a given λ in equation 1. The Lagrange rate-distortion cost function is given below:

$$\min_{m} J(m) = D(m) + \lambda.R(m) \tag{1}$$

Here, λ is the Lagrange multiplier, m $\in$ M, M is the set of all possible parameter combinations, R is the rate, D is the distortion which is commonly known as the sum of squared errors and J is the rate-distortion (RD) cost. The complexity of RDO process totally, depends on M and there are more than 75,000 parameter combinations are possible for a root level coding unit (64 $\times$ 64).

To reduce the time complexity in encoder side, many guidelines are suggested in different ways. A similar group of research is already proposed to minimize the coding units complexity by using smart-based conditions [4, 5]. Skipping some mode tests of individual nodes (CUs), at the time of RDO process is a way to speed up the encoder. These procedure is known as a fast CU splitting process and this type of research [6, 7, 8] is already included in HEVC reference software. Another way to speed up the encoder by using machine learning approach and this approach has succeeded in most of the cases. Heindel et al. [9] and Zhang et al. [10] proposed a fast CU split decision by using SVM (Support Vector Machine) classifier. Heindel showed that 60% time complexity is saved on average, whereas 3-4% bit rate is increased by using SVMs for fast splitting the CUs in [9]. A similar study is experimented by Westerland, and they used the decision tree (D-Tree) classifier instead of SVM. This study is very recent and effective for the reduction of complexity. Westerland et al. [11] showed that 0.7% bit rate is increased, whereas 42.1% encoding time is reduced. The success of the authors of these papers by using machine learning approaches motivated us in this research area.

In this paper, we presented a method for splitting a CU or not at the time of rate-distortion optimization (RDO) process. Our model uses the decision tree classifier for classification purpose. The authors of [11] also uses decision tree but different features. Our model uses only two features and has a high accuracy for predicting the blocks or CUs where the uniform motion has or not and traditionally as less as features, needs a short time for classification. All two features of a CU are collected before the RDO process is started. If our model has a prediction that a CU has only a single motion, then this CU is skipped for further partitioning and RDO process is early terminated, otherwise this CU is handover-ed to HEVC for further partitioning. In this way. the time complexity is saved. Latest video codec such as versatile video coding (VVC) which is finalized recently, has no such fast split decision as HEVC environment is used there.

The rest of the paper is organized as follows: in section 2, we explain how the dataset is generated for classification purpose. In section 3, classification process is delineated in briefly. We show the experimental results in section 4. Finally, the paper is concluded in section 5.

2. DATASET GENERATION

In the case of machine learning algorithm, it needs lots of data for training and testing to make a prediction model. Luckily there is a lot of data which are available in the various test video sequences. In the following subsection, how the dataset is formed is described in details.

2.1. Feature Engineering

Features (commonly known as attributes) are used for classification purposes and feature engineering or feature selection is an essential part for designing any kind of machine learning approach. The features are generated from the displayed frame image in this research as the residual image is available in encoder side. Displayed frame difference (DFD) or residual image can be determined by subtracting the motion-compensated prediction frame from the current frame. The two features namely error energy E_T and low frequency energy E_L for any N $\times$ N block can be defined as:

$$E_T = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} d^2(x, y) \tag{2}$$

$$E_L = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1-u} F_{dct}^2(u, v) \tag{3}$$

Here, d(x,y) is a residual image in the spatial domain, whereas $F_{dct}(u, v)$ is a discrete cosine transform (DCT) co-efficient in the frequency domain. E_T refers to the summation of error energy of a block and E_L refers to the summation of upper half triangle DCT co-efficients of an error image block. The reason for choosing the upper half triangle co-efficients is that most of the coefficients in the lower half triangle are zero. So, these co-efficients can be neglected while preserving almost the same accuracy and the benefit is reducing the computational time into half, because there is no need to calculate the lower half triangle coefficients. Traditional codec fails for motion compensation if the block contains multiple motions. The reason for choosing the feature E_T is that if the block has motion discontinuity then the error energy of that block is high in tradition. Similarly, any failure of motion compensation in motion discontinuous region has a significant reflection in the frequency domain and for this reason E_L is chosen. The significance of the attributes can be shown by drawing ROC curve. Fig. 3 shows the significance of the feature E_T and E_L. True positive rate (TPR) refers to the fraction of multiple

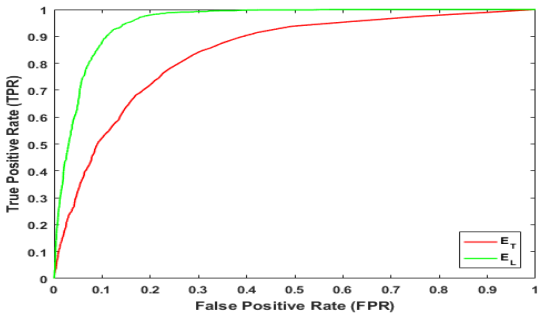

Fig. 3. Receiver operator characteristics (ROC) curve of the feature E_T and E_L, generated from BasketballDrive sequence (1080p).

motion blocks successfully identified by a feature. False positive rate (FPR) refers to the fraction of single motion blocks wrongly classified by a feature.

The ground truth value of a block is used as the class level of that block. The blocks which contain multiple motions are leveled as positive class and others are leveled as negative class. So, in summary every N $\times$ N block of a residual image has two features and corresponding class level and this information is treated as a single record or row of the dataset. Four various standard JCT-VC test sequences listed in Table 1, are used for generating the dataset. In this way, the dataset is prepared.

Table 1. The test sequence name, resolution, frame rate from the JCT-VC [12] used in this research.

Sequence Name	Resolution	Frame Rate
$BasketballPass$	416×240	50
$RaceHorses$	416×240	30
$BasketballDrive$	1920×1080	50
$Kimono$	1920×1080	24

3. CLASSIFICATION PROCESS

The complexity of spiting a CU or not totally, depends on how the classifier classifies the CU block more accurately. The full procedure of classification process is described in the following three subsections.

3.1. Dataset Preprocessing

The generated dataset has tow significant features and corresponding class label that is explained elaborately in section 2. After examining the dataset, it is seen that the ratio of negative class and positive class tuples is 1.89, i.e. negative class has almost two times tuples than the positive class and this phenomenon is commonly called class imbalance problem. If

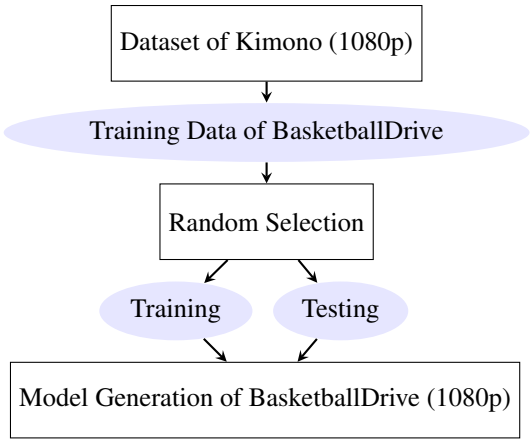

Fig. 4. Example of Model generation for BasketballDrive sequence from the dataset of Kimono sequence.

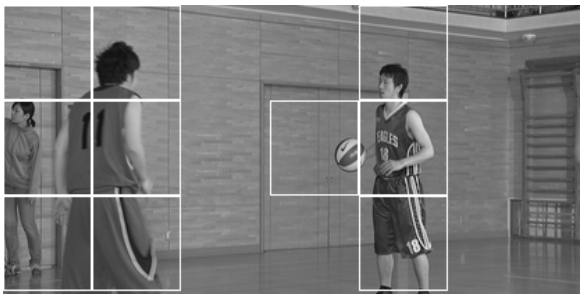

Fig. 5. Prediction of frame number 4 of BasketBallPass sequence (240p). White boundary blocks (64 $\times$ 64) are classified as multiple motion blocks by using the proposed classifier and rest of the blocks or regions are deemed to as containing single motion.

a dataset has class imbalance problem then the output accuracy is not reliable because the the algorithm has bias problem towards the major class by applying any kind of machine learning approach. To obtain a good and reliable result class imbalance problem must be solved. The authors of this paper used the oversampling technique for solving this issue [13]. Another issue regarding the dataset is that the feature values are not scaled. To speed up the classification process or to converge more rapidly, all features of the dataset are scaled up into [0,1] range.

3.2. Classifier Selection

Decision tree is used for classifying the motion discontinuity of the blocks. The reason behind using the decision tree over SVM is that the prediction of motion discontinuity can be obtained more easily. This algorithm is often known as white box compared to other machine learning algorithms, which are referred as a black box. Breiman et al. [14] first developed the classification and regression tree in 1984. Decision tree algorithm splits the node based on impurity or error of the

Table 2. Detail Experimental Results of All Test Sequence.

Sequence Name	Sensitivity (%)	Specificity (%)	Precision (%)	Recall (%)	F_1 Score (%)	Accuracy (%)
$BasketballPass$ $(240p)$	91.52	87.69	88.13	91.52	89.79	89.60
$RaceHorses$ $(240p)$	75.78	89.76	74.23	75.78	75.00	82.77
$BasketballDrive$ $(1080p)$	93.95	86.33	87.34	93.95	90.53	90.15
$Kimono$ $(1080p)$	79.85	80.26	78.01	79.85	77.93	80.05
Average	**85.26**	**86.01**	**81.93**	**85.26**	**83.31**	**85.64**

node. Several split criteria are used for this algorithm. For example, gini index (GI), information gain (IG) or deviance and towing are commonly used in this purpose [13]. The authors of this paper used deviance for spiting a certain node and the formula of deviance or IG of a node x is as follows:

$$Deviance\ (x) = -\sum_{i}\ p(i)\ *\ log\ p(i) \qquad (4)$$

Here, the sum is over the classes i at the node, and p(i) is the observed fraction of classes with class i that reach the node. A pure node has deviance 0 otherwise, it has a positive value.

3.3. Classification

To generate a model for classification, it is needed to train the dataset at first. For minimizing the bias problem, one sequence data is trained for testing another sequence of the same resolution. Fig. 4 shows the process, how the model is generated for BasketballDrive (1080p) sequence. Training data of BasketballDrive sequence are randomly partitioned into two categories training and testing. The authors of this paper tested three times for evaluating the model by keeping 10%, 15%, 20% data respectively for testing (i.e. cross validation dataset) and remaining data are used for training (off-line). Holdout method is used for cross-validation and pruning is used to minimize the over-fitting problem. Once the model is created for a particular sequence, then that sequence dataset is used for testing and a prediction scheme is shown in fig. 5. The average results for various metrics that's are commonly used for classifier performance are shown in Table 2. Similar strategy is used for other sequences for creating the model.

4. EXPERIMENTAL RESULTS

The experimental section is divided into two subsections. In subsection 4.1, the experimental setup is described. Subsection 4.2 gives a summary of the experimental results.

4.1. Experimental Setup

The proposed method is implemented in HEVC reference software. We used HM - 16.20 [15] for practical implementation and four QP values (22, 27, 32, 37) used to encode each sequence thats were used for experimental analysis. Low delay main GOP structure, i.e. IPPPP.....P was used as a configuration file.

4.2. Performance Analysis

For performance analysis, at first 50 frames of each sequence were encoded by the unmodified version of HM - 16.20 with maintaining the common test conditions of HEVC reference software [12] and the encoding results were recorded. Secondly, the same 50 frames were encoded by the proposed HM - 16.20 encoder (modified version). The obtained two results were compared by using Bjøntrgaard delta bitrates piece-wise cubic interpolation method [16] and in Table 3 all encoding results are summarized.

Table 3. Summery of encoding results.

Sequence Name	Δ Rate	Δ Enc. Time
$BasketballPass$ $(240p)$	0.92%	-43.59%
$RaceHorses$ $(240p)$	1.89%	-47.28%
$BasketballDrive$ $(1080p)$	0.98%	-42.97%
$Kimono$ $(1080p)$	1.73%	-46.23%
Average	**1.38%**	**– 45.02%**

5. CONCLUSION

Machine learning approach plays a vital role in fast partitioning the CUs at the encoder side, when rate-distortion optimization process is invoked. In this paper, we proposed D-Tree classifier which used fruitful features for fast CU split decision to speed up the encoder. By using machine learning algorithm so-called decision tree algorithm, on average 45.02% time complexity is reduced whereas, on average 1.38% bit rate is increased over standalone HEVC. For BasketballPass and BasketballDrive sequences only 0.92% and 0.98% bit rate is increased respectively. On the other hand, 1.89% and 1.73% bit rate is increased for RaceHorses and Kimono sequence respectively due to having a rapid change of camera motion and illumination problem compared to other sequences and for this reason our proposed classifier performance is less compared to BasketballPass and BasketballDrive sequences. In future, we will try to increase the performance of the classifier for the test sequences.

6. REFERENCES

[1] Gary J Sullivan, Jens-Rainer Ohm, Woo-Jin Han, Thomas Wiegand, et al., "Overview of the high efficiency video coding(hevc) standard," *IEEE Transactions on circuits and systems for video technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[2] "Alliance for Open Media, "AOMedia Video 1 (AV1) 2018"," [Online], Available: https://aomedia.org/.

[3] Il-Koo Kim, Junghye Min, Tammy Lee, Woo-Jin Han, and JeongHoon Park, "Block partitioning structure in the hevc standard," *IEEE transactions on circuits and systems for video technology*, vol. 22, no. 12, pp. 1697–1706, 2012.

[4] Jarno Vanne, Marko Viitanen, and Timo D Hämäläinen, "Efficient mode decision schemes for hevc inter prediction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 9, pp. 1579–1593, 2014.

[5] Andreas Heindel and André Kaup, "Fast intra mode decision in hevc using early distortion estimation," in *2015 IEEE China Summit and International Conference on Signal and Information Processing (ChinaSIP)*. IEEE, 2015, pp. 559–563.

[6] Kiho Choi, Sang-Hyo Park, and Euee S Jang, "Coding tree pruning based cu early termination," *JCT-VC document, JCTVC-F092*, 2011.

[7] Ryeong-hee Gweon and Yung-Lyul Lee, "Early termination of cu encoding to reduce hevc complexity," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 95, no. 7, pp. 1215–1218, 2012.

[8] J Yang, J Kim, K Won, H Lee, and B Jeon, "Early skip detection for hevc," *document JCTVC-G543*, 2011.

[9] Andreas Heindel, Thomas Haubner, and André Kaup, "Fast cu split decisions for hevc inter coding using support vector machines," in *2016 Picture Coding Symposium (PCS)*. IEEE, 2016, pp. 1–5.

[10] Yun Zhang, Sam Kwong, Xu Wang, Hui Yuan, Zhaoqing Pan, and Long Xu, "Machine learning-based coding unit depth decisions for flexible complexity allocation in high efficiency video coding," *IEEE Transactions on Image Processing*, vol. 24, no. 7, pp. 2225–2238, 2015.

[11] Natasha Westland, André Seixas Dias, and Marta Mrak, "Decision trees for complexity reduction in video compression," in *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019, pp. 2666–2670.

[12] Frank Bossen et al., "Common test conditions and software reference configurations," *JCTVC-L1100*, vol. 12, 2013.

[13] Jiawei Han, Jian Pei, and Micheline Kamber, *Data mining: concepts and techniques*, Elsevier, 2011.

[14] L Breiman, J Friedman, CJ Stone, and RA Olshen, "Classification and 715 regression trees, the wadsworth and brooks-cole statistics-probability series," 1984.

[15] "HM Reference Software for High Efficiency Visdeo Coding (HEVC)," [Online], Available: https://hevc.hhi.fraunhofer.de/.

[16] Gisle Bjontegaard, "Calculation of average psnr differences between rd-curves," *VCEG-M33*, 2001.