IEICE TRANS. INF. & SYST., VOL.E101–D, NO.9 SEPTEMBER 2018

LETTER *Special Section on Picture Coding and Image Media Processing*

Fast CU Termination Algorithm with AdaBoost Classifier in HEVC Encoder

Yitong LIU[†a)], **Wang TIAN**[†], **Yuchen LI**[†], ***Nonmembers***, ***and*** **Hongwen YANG**[†], ***Member***

SUMMARY High Efficiency Video Coding (HEVC) has a better coding efficiency comparing with H.264/AVC. However, performance enhancement results in increased computational complexity which is mainly brought by the quadtree based coding tree unit (CTU). In this paper, an early termination algorithm based on AdaBoost classifier for coding unit (CU) is proposed to accelerate the process of searching the best partition for CTU. Experiment results indicate that our method can save 39% computational complexity on average at the cost of increasing Bjontegaard-Delta rate (BD-rate) by 0.18.

key words: *HEVC, fast algorithm, early termination, AdaBoost*

1. Introduction

High Efficiency Video Coding (HEVC) improves compression performance comparing with H.264/AVC in the range of 50% bitrate reduction for equal perceptual video quality [1]. HEVC splits a frame into coding tree units (CTUs) that may be split into coding units (CU) recursively. HEVC will search all the possible splitting patterns, then the best pattern will be used to compress the CTU. The quadtree structure in CTU splitting contributes to huge promotion in codec performance and therefore brings heavy computational burden [2].

Numerous works focused on speeding up quadtree pruning in HEVC. [3] proposed a machine learning based CU depth decision algorithm with given rate-distortion (RD) cost constraints. [4] proposed a CU size decision algorithm for intra coding. The correlation of neighboring CUs' depth, distribution of RD cost, and distribution of residual data were analyzed, respectively. [5] considered CU splitting or non-splitting as a binary classification problem. The authors proposed a CU size decision algorithm based on probability statistics. [6] proposed a CU split decision algorithm for inter coding based on support vector machine (SVM) where a different SVM for each depth level was employed.

In this paper, we propose a method to perform early termination for redundant CU structure searching process with AdaBoost classifier. The rest of this paper is organized as follows. Section 2 introduces the process of feature filtering. The AdaBoost model is built in Sect. 3. The overall performance is demonstrated in Sect. 4.

Manuscript received December 15, 2017.
Manuscript revised April 12, 2018.
Manuscript publicized June 20, 2018.

[†]The authors are with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, No.10 Xitucheng Road, Haidian District, Beijing, 100876 China.
a) E-mail: liuyitong@bupt.edu.cn

DOI: 10.1587/transinf.2017PCL0001

2. Feature Selection with Decision Trees

In our works, the CU splitting or non-splitting is modeled as a binary classification problem. A feature set is built before feeding features to AdaBoost classifier. The constituent parts of the feature set are listed in Table 1. Eight sequences are chosen from [7] as the training sequences and listed in Table 2.

RD cost proved significant in making the splitting decision [3]–[6]. Specifically, RD cost (J) is calculated [8] by

$$J = D + \lambda \cdot R \tag{1}$$

where λ is the Lagrangian multiplier, D and R are the distortion and bitrate for the sequence.

However, RD cost is related to the quantization parameter (QP) value. To explore the relationship between RD cost and QP value, training sequences in Table 2 are encoded with different QP value and the average RD cost of splitting CUs and non-splitting CUs are recorded, respectively. Figure 1 gives partial results for illustration. The RD

Table 1 Potential factors affecting the CU splitting

Feature Name	Description
VAR_SUM_CU MEAN_SUM_CU MAX_ SUM_CU	(Variance, Mean, Max value) of sums of four sub-CUs' residual coefficient.
VAR_NZ_CU MEAN_NZ_CU MAX_NZ_CU	(Variance, Mean, Max value) of nonzero coefficient ratio in four sub-CUs.
AVE_SUM_PU	Average of sums of residual coefficient in each predicting unit (PU).
MAX_NZ_PU	Max value of nonzero ratio of residual coefficient in each PU.
RDcost	The result of Lagrangian multiplier method.

Table 2 Training sequences

Class A	PeopleOnStreet
Class B	BQTerrace
	ParkScene
Class C	BasketballDrill
	BQMall
Class D	BasketballPass
	RaceHorses
Class E	FourPeople

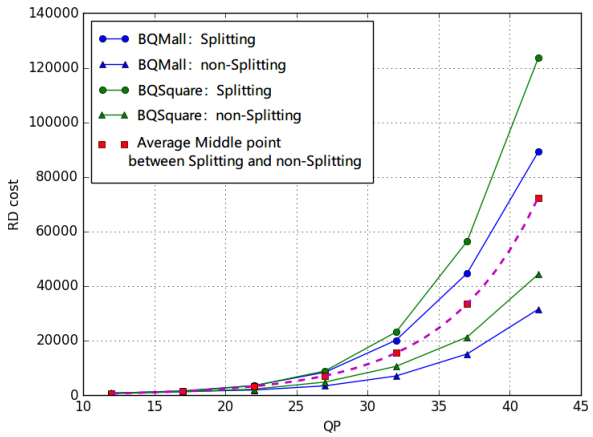

Fig. 1 Relationship between RD cost and QP value. Two sequences (*BQMall*, *BQSquare*) are tested using QP values of {12, 17, 22, 27, 32, 37, 42}.

cost increases as the QP value raises monotonously. The curves marked by dots and triangles respectively represent two types of processing of the CU: "splitting" and "non-splitting". "splitting" means the current CU fails to achieve the minimum RD cost on its current size and needs to be split into four sub-CUs to conduct pattern searching, respectively. "non-splitting", on the contrary, indicates that the current CU can obtain best RD cost without splitting into sub-CUs. The red squares representing the average RD cost of both sequences at different QP values, separate whether the CU is split or not. As Fig. 1 shown, the red squares can be well fitted to an exponential function

$$f(QP) = \alpha \cdot e^{\beta \cdot QP} \tag{2}$$

where α and β are constant coefficients to be determined. Furthermore, the original RD cost can be adjusted into a suitable parameter for training model by

$$J_{offset} = J - \alpha \cdot e^{\beta \cdot QP} \tag{3}$$

where J is the original RD cost defined in Eq. (1), J_{offset} means the delta of original RD cost and the partition curve as illustrated in Fig. 1.

With the data gained from encoding the training sequences in Table 2, the value of (α, β) is regressed. For CU size 16×16 the value of (α, β) is (120.21, 0.15) and (460.12, 0.14) for CU size 32×32. RD cost offset proves to be an effective feature and therefore added to our feature set.

For the sake of the simplicity, a decision tree is used to filter the candidate features. An experiment is designed to filter our feature set. Firstly, video sequences are compressed with original HEVC encoder. The decision of codec in every CUs and its correlative features are recorded. Eight training sequences in Table 2 are test with the *random_access_main* profile. About 1 million records are collected for further analysis. Secondly, a dataset is built with data collected and half of the dataset is taken as training data to build the decision tree. Thirdly, the feature and entropy decreasing margin in each node of the tree are recorded after building the decision tree. The entropy decreasing margin indicates the importance of the features. The second and the third step are repeated 100 times. Accroding to the result listed in Table 3, *RD cost offset*, *VAR_SUM_CU*, and *VAR_NZ_CU* are finally selected as the input in our proposed AdaBoost model.

Table 3 Entropy decreasing ratio

Feature Name	Entropy Decreasing Ratio (%)
RD cost offset	88.03
VAR_SUM_CU	7.43
VAR_NZ_CU	1.97
MEAN_NZ_CU	1.00
MEAN_SUM_CU	0.87
MAX_NZ_CU	0.49
MAX_SUM_CU	0.17
MAX_NZ_PU	0.00
MAX_NZ_PU	0.00

3. Fast Quadtree Algorithm with AdaBoost Model

The AdaBoost (adaptive boosting) algorithm is proposed [9] as a general method for generating a strong classifier out of a set of weak classifiers. A decision tree is used as a base classifier in our model. As a weak classifier, the height of the base decision tree is only one. At the beginning, AdaBoost chooses the best decision tree learners for the current sample set. After a weak classifier is added to AdaBoost, the weight of "wrong" samples is raised. In the next classification, AdaBoost will pay more attention to the "wrong" samples until the number of classifiers reaches its limit. When the AdaBoost model is built, the final prediction comes from the weighted combination of all the weak classifiers.

In our proposed method, AdaBoost classifier predicts the probability of splitting before further searching. A greater probability of splitting means that the further splitting and pattern searching for the current CU are more likely to achieve better rate distortion performance. If the probability of splitting is lower than the threshold, the quadtree structure searching terminates. Otherwise, the further searching is executed.

Before training our model, the number of weak classifiers should be determined. The model performance of using different numbers of weak classifiers is evaluated with the Area Under Curve (AUC) [10]. When the value of AUC is greater than 0.5, the model is considered to be of use for prediction if appropriate thresholds are selected. As shown in Fig. 2, the value of AUC is in (0.84, 0.90) indicating that the model has good performance, especially when the number of classifiers exceeds 4. More classifiers bring better performance except for certain cases. When the number of weak classifiers is 12, 16, 20, 22, 26 or 28, the model performance decreases slightly. However, model performance improvement is negligible when the number of weak classifiers exceeds a certain number (around 5). To balance model performance and computational complexity, the number of classifiers is 6 in our proposed AdaBoost model.

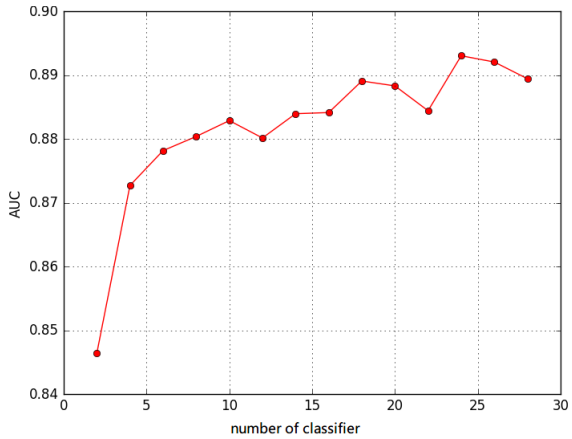

Fig. 2 The relationship between the number of weak classifier and AUC.

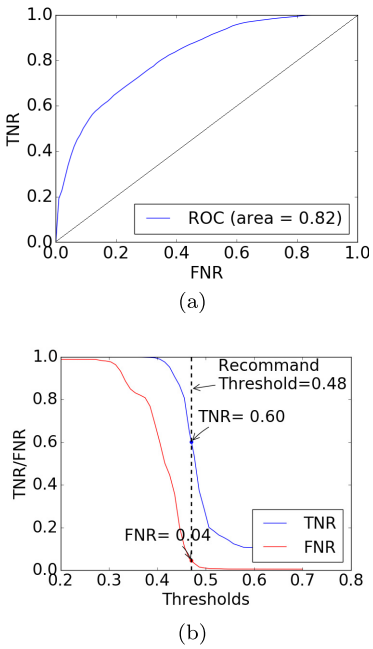

Fig. 3 ROC curve is illustrated in (a). The relationship between threshold and TNR/FNR is showed in (b).

When selecting CU splitting threshold, two types of errors in CU splitting decision are considered. One occurs when a CU actually needs splitting while the AdaBoost terminates further searching. The other occurs when CU does not need splitting while the AdaBoost actually does. The former brings worse result than the latter because a premature termination of CU pattern searching caused by the former error leads to quality degradation. Unnecessary CU pattern searching caused by the latter merely increases computational complexity. However, eliminating the classifying error is impossible. A compromise way is to set a proper threshold deciding the result of classification.

Figure 3 illustrates relationship between false negative rate (FNR), true negative rate (TNR), and the decision threshold. FNR represents the first error which means a CU needs splitting while the AdaBoost terminates further searching. (1-TNR) represents the second error which means CU dose not need splitting while the AdaBoost continues. The smaller FNR value or the greater TNR value is, the more accurate output of the classifier provides. Considering the coding performance degradation caused by these two errors, FNR is chosen to decide the threshold for early termination of CU splitting.

Figure 3(a) shows Receiver Operating Characteristic Curve (ROC) [11] which is often used for evaluating the performance of a classifier. The diagonal represents the random guessing model (area = 0.5). That the carve is above the diagonal indicates that the proposed model works well Fig. 3(b) shows the FNR value and the TNR value when the threshold is 0.48. In our method, the output of AdaBoost model is probability of "splitting" a CU. When the probability is below the threshold, further searching is terminated. The proposed method is not used with CU size 64×64 to avoid obvious quality degradation when mistakes are committed by the classifier.

4. Experiment Results

The proposed method is implemented with scikit-learn [12] and verified on HM10. The test conditions are defined in

Table 4 Overall performance of proposed method

Test Sequences		*TS*(%)	*BD* − *rate*(%)	*TS*(%)	*BD* − *rate*(%)	*TS*(%)	*BD* − *rate*(%)
		Thresholds					
		0.49		0.48		0.47	
Class A	NebutaFestival	33%	0.15	6.5%	0.01	1%	-0.03
	SteamLocomotiveTrain	62%	1.77	52.3%	0.14	46%	0.08
	Traffic	62%	8.58	49.4%	0.32	43%	0.13
Class B	BasketballDrive	59%	5.26	41.8%	0.16	37%	0.02
	Cactus	62%	7.11	38.6%	0.24	33%	0.07
	Kimono1	63%	3.12	47.7%	0.19	42%	0.12
Class C	PartyScene	48%	4.77	18.0%	0.19	12%	0.04
	RaceHorsesC	48%	7.00	25.0%	0.20	19%	0.03
Class D	BlowingBubbles	52%	7.00	19.8%	0.24	14%	0.09
	BQSquare	58%	6.61	24.8%	0.11	19%	0.05
Class E	Johnny	66%	3.88	63.2%	0.02	61%	-0.02
	KristenAndSara	65%	5.20	61.4%	0.13	58%	0.00
	vidyo1	63%	4.83	61.9%	0.36	60%	0.15
Average result		**57%**	**5.02**	**39%**	**0.18**	**34%**	**0.06**

[7] and *random_access_main* profile is used for compression. The overall results of test sequences are listed in Table 4. Each sequence is tested with QPs at 22, 27, 32, 37, and Bjontegaard-Delta rate (BD-rate) loss is calculated to provide a comparable result. The average encoding time saving (TS) is calculated by

$$TS\,(\%) = \frac{T_{HM10} - T_{PROP}}{T_{HM10}} \times 100\%. \quad (4)$$

Notably, the sequences listed in Table 2 for training model are not used for result verification.

The time reduction narrows and the encoding quality improves as the threshold decreases. When the threshold is 0.49, the encoding quality degrades notably, especially for some specific sequences *(Traffic, Cactus)*. When the threshold is 0.47, the qualities of some sequences are even superior to original codec *(NebutaFestival, Johnny)*. When the threshold is 0.48, the time reduction is about 39% with negligible quality degradation.

5. Conclusion

In this paper, an early termination algorithm for CU structure deciding in HEVC encoding process is proposed. An AdaBoost classifier is applied to predict CU splitting probability according to our feature set before further searching. If the probability of splitting is below the threshold, the searching process is terminated immediately to reduce encoding time. Experiment results show that the time reduction is about 39% at the cost of 0.18% BD-rate increasing.

References

[1] G.J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (hevc) standard," IEEE Trans. Circuits Syst. Video Technol., vol.22, no.12, pp.1649–1668, Dec. 2012.

[2] J.R. Ohm, G.J. Sullivan, H. Schwarz, T.K. Tan, and T. Wiegand, "Comparison of the coding efficiency of video coding standards—including high efficiency video coding (hevc)," IEEE Trans. Circuits Syst. Video Technol., vol.22, no.12, pp.1669–1684, Dec. 2012.

[3] Y. Zhang, S. Kwong, X. Wang, H. Yuan, Z. Pan, and L. Xu, "Machine learning-based coding unit depth decisions for flexible complexity allocation in high efficiency video coding," IEEE Trans. Image Process., vol.24, no.7, pp.2225–2238, July 2015.

[4] T. Fan, G. Wang, and X. Shang, "Fast coding unit size decision in hevc intra coding," IEICE Trans. Inf. & Syst., vol.E99-D, no.7, pp.1953–1956, July 2016.

[5] X. Jiang, T. Song, W. Shi, T. Katayama, T. Shimamoto, and L. Wang, "Fast coding unit size decision based on probabilistic graphical model in high efficiency video coding inter prediction," IEICE Trans. Inf. & Syst., vol.E99-D, no.11, pp.2836–2839, Nov. 2016.

[6] A. Heindel, T. Haubner, and A. Kaup, "Fast cu split decisions for hevc inter coding using support vector machines," Picture Coding Symposium (PCS), 2016, pp.1–5, IEEE, 2016.

[7] F. Bossen and H. Common, "Test conditions and software reference configurations, jct-vc doc," L1100, Jan, 2013.

[8] G.J. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," IEEE Signal Process. Mag., vol.15, no.6, pp.74–90, Nov. 1998.

[9] Y. Freund and R.E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," J. Comput. Syst. Sci., vol.55, no.1, pp.119–139, 1997.

[10] A.P. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," Pattern Recognit., vol.30, no.7, pp.1145–1159, July 1997.

[11] J.A. Hanley and B.J. McNeil, "The meaning and use of the area under a receiver operating characteristic (roc) curve," Radiology, vol.143, no.1, pp.29–36, 1982.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in python," J. machine learning research, vol.12, no.Oct, pp.2825–2830, 2011.