DOI: 10.2352/ISSN.2470-1173.2016.2.VIPC-236

Machine Learning-based Early Termination in Prediction Block Decomposition for VP9

Xintong Han; University of Maryland; College Park, MD. Yunqing Wang; Google Inc.; Mountain View, C A. Yaowu Xu; Google Inc.; Mountain View, CA. Jim Bankoski; Google Inc.; Mountain View, CA.

Abstract

VP9 is an open-source video codec released by Google. It introduces superblocks (SBs) of size 64×64*, and uses a recursive decomposition scheme to break them all the way down to* 4×4 *blocks. This provides a large efficiency gain for VP9. However, it also brings large computational complexity when encoding because of the rate distortion (RD) optimization on prediction blocks. This paper proposes a method that can early terminate the block partitioning process based on the information of the current block. We first model the early termination decision as a binary classification problem. Second, to solve this classification problem, a weighted linear Support Vector Machine (SVM) is trained whose weights are determined by the RD cost increase caused by misclassification. Finally, we model the parameter selection of the SVM as an optimization problem, which can enable us to control the trade-off between time saving and RD cost increase. Experimental results on standard HD data shows that the proposed method can reduce the complexity of partitioning prediction blocks while maintaining comparable coding performance - The Bjøntegaard delta bit rate is* $\sim$*1.2% for* $\sim$*30% encoding time reduction.*

Introduction

Google finalized a next generation open-source video codec called VP9 [1]. In this codec, superblocks (SBs) with size 64×64 can be recursively decomposed all the way down to 4×4 blocks as shown in Figure 1. This scheme provides a large quality gain but also brings some computational complexity increase because of the RD optimization on the prediction blocks. Thus, it is important to reduce the encoding time of this process without sacrificing too much quality. One of the most popular methods is to early terminate the block partitioning process based on the information of current block [2, 3]. In this paper, we model the early termination decision as a binary classification problem and use weighted Support Vector Machine (SVM) to solve the problem.

The main contributions of this paper include:

1. Propose a novel weight design and optimal parameter selection method for the weighted SVM classifier which solves the early termination task.

2. Use linear SVM which is much faster and easy to apply in the current codec.

3. To the best of our knowledge, this is the first paper that uses machine learning based method to reduce the computational complexity of the VP9 encoder.

4. The method presented may be easily extended to other codecs that use RD cost to decide the block partitioning by choosing similar features.

Related Work

Many machine learning based methods have been used to do Coding Unit (CU) decisions or early termination for HEVC.

Shen and Yu [2] used weighted SVMs to do CU early termination for HEVC standard. The weights in the weighted SVMs are decided by the RD loss due to misclassification. They also provided a feature selection scheme based on the F-scores to find the good features. However, the weight of a block in their method is not well-normalized by the characteristics of the video that block belongs to, which makes the method not generalized for the task.

To reduce the computational complexity of CU depth decisions, Zhang et al. [4] built a joint classifier considering the risk of false prediction, and proposed an optimal parameter determination algorithm trading RD cost degradation for time reduction. However, this method treats every sample equally, which does not work well since misclassification of samples gives different RD cost degradation.

There have been also many related works dealing with reducing the complexity CU partitioning based on non-machine learning approaches. Shen et al. [5] used the information of the previous frame and neighboring CUs to determine the CU depth range and skip some specific depth levels rarely used. Kim et al. [3] early terminated the CU splitting based on an adaptive threshold that depends on the RD cost of neighboring CUs and CUs of higher depth. In the work of Gweon and Yung-Lyul [6], the RD cost difference between the root and children CUs in a CU quad-tree is used to select the CU depth. However, these methods use less rich features (usually 2-3 features), and do not leverage other information in training videos, thus do not work better than

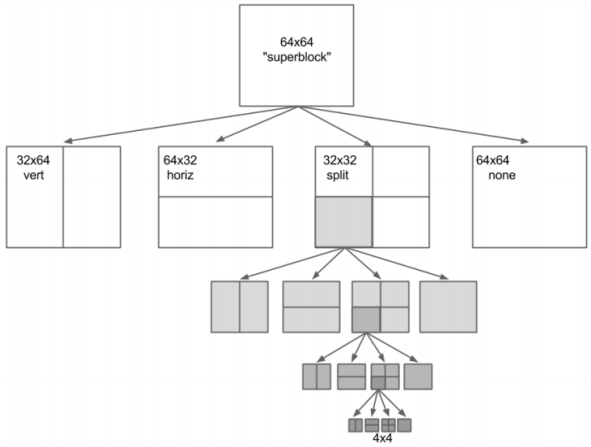


Figure 1: Recursive SB partitioning structure in VP9

©2016 Society for Imaging Science and Technology
DOI: 10.2352/ISSN.2470-1173.2016.2.VIPC-236

machine learning based methods.

Moreover, due to intrinsic difference between HEVC and VP9, most of the methods cannot be directly applied to VP9 encoder. To this end, we propose an early termination method for prediction block splitting process in VP9 based on machine learning techniques, which tackles the early termination with a novel weight selection method and a way to get the optimal parameters for training the classifiers.

Proposed method

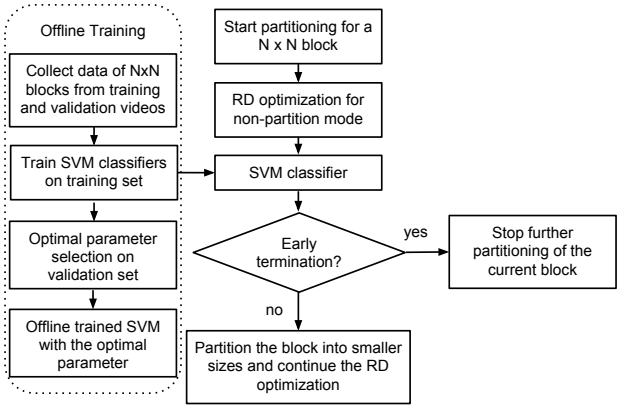


Figure 2: Framework of the algorithm.

We propose a method to early terminate the block splitting process for INTER frames in VP9 encoder. The overview of our method is shown in Figure 2. The whole approach consists of two main parts: training (enclosed in the dashed rectangle in Figure 2), and testing. In the training part, features that are useful for determining further block partitioning are extracted from training and validation videos. Then, we treat the blocks with non-partition mode (none in Figure 1) as positive samples and the other partition modes (horizontal, vertical, and splitting partition in Figure 1) as negatives to build a binary weighted SVM. Meanwhile, a parameter selection algorithm is proposed with which people can decide to trade off how much quality loss for encoding time reduction. Furthermore, we also use a parameter selection algorithm to select the optimal parameter for the SVM based on the RD cost increase of misclassification. Note that to early terminate a $N \times N$ block, the SVM classifier should depend on the value of N. For VP9 codec, N can be 64, 32, 16, and 8. We choose 3 of the possible Ns - 64, 32, 16, since blocks smaller than 8×8 blocks are rarely used and cannot not provide useful features for classification due to its smaller size. As a result, 3 binary classifiers are trained and applied to test videos. When testing, we first run the RD optimization for the non-partition mode, and the resulting features are used as input to the offline trained SVM. If this block is classified to be no further partitioned, the partitioning process comes to an end; otherwise, other partitioning methods will be tried and the one with lowest RD cost will be selected.

The details of our method are described in the following subsections.

Feature

The following features of INTER blocks are used in the early termination algorithm:

1) x_{rate}: Rate cost of the non-partition mode in the current block.

2) x_{dist}: Distortion cost of the non-partition mode in the current block.

3) x_{motion}: Magnitude of the motion vector of the non-partition mode in the current block. We define this feature as $(|MV_x| + |MV_y|)/2$ in our method, where MV_x and MV_y are the motion vector in x- and y- directions, respectively.

4) Context partitioning information. For the current block, the context information includes the partitioning of its co-located block in last frame x_{last_part}, the above block x_{above_part}, and the left block x_{left_part}. For example, for a block of size 32×32, if the co-located block is inside a larger block (64×64, 32×64, or 64×32), $x_{last\_part} = 0$; if the co-located block is a non-partition block with the same size of 32×32, $x_{last\_part} = 1$; otherwise, the co-located block is further partitioned to blocks of small sizes, $x_{last\_part} = 2$. The values of x_{left_part} and x_{above_part} are decided in the same fashion. Finally, we use x_{last_part} as the context information of last frame denoted as $x_{last\_context}$, and $(x_{above\_part} + x_{left\_part})/2$ as the context of current frame denoted as $x_{current\_context}$.

5) x_{eobs}: Number of nonzero coefficients for encoding the non-partition mode in the current block, i.e., the sum of eobs (end of blocks).

6) x_q: Q value of the current frame.

As a result, the features of a block form a vector $\mathbf{x} = [x_{rate}, x_{dist}, x_{motion}, x_{last\_context}, x_{current\_context}, x_{eobs}, x_q]$ to represent it.

Note that these features need to be normalized before training the classifier. We use a dimension-wised softmax normalization method to achieve this goal:

$$x_i' = \frac{1}{1 + e^{-\left(\frac{x_i - \mu_i}{\sigma_i}\right)}} \tag{1}$$

where x_i is the i-th dimension of the original feature vector, μ_i and σ_i are the mean and standard deviation of that dimension in training data, respectively. This normalization uses a sigmoid function that can transform data into the interval (0,1) and reduce the influence of extreme values or outliers without removing them.

Classifier

To speed up the training and testing process of the early termination, we use a linear SVM [7] classifier instead of nonlinear one for several reasons: First, running nonlinear classifiers during testing can bring a much larger overhead than linear ones. Also, based on our observation, nonlinear and linear SVMs give the similar performance in terms of classification accuracy. Plus, it is easier to plug the linear model in the codec, and more feasible for hardware speed-up.

The main idea of SVM classifier is to find a hyperplane that can separate the training samples of different classes while maximize the margin between these classes. Given N_1 positive samples and N_0 negative samples $\{\mathbf{x}_i, y_i\}$, $i = 1, 2, ..., N_0 + N_1$, where $\mathbf{x}_i$ is the feature vector of the i-th sample. $y_i = 1$ for positives and -1 for negatives. A weighted linear SVM minimizes the cost function in

DOI: 10.2352/ISSN.2470-1173.2016.2.VIPC-236

(2).

$$J(\mathbf{w},b,\xi) = \frac{||\mathbf{w}||^2}{2} + \sum_{i=1}^{i=N_0} W_i\xi_i + \sum_{i=N_0+1}^{i=N_0+N_1} W_i\xi_i$$

s.t.

$$(\mathbf{w}^T\mathbf{x}_i + b)y_i \geq 1-\xi_i, \xi_i \geq 0 \tag{2}$$

where W_i is the weight for instance $\{\mathbf{x}_i, y_i\}$, if W_i is large, the cost of misclassifying $\{\mathbf{x}_i, y_i\}$ will be high. ξ is a vector of parameter ξ_i.

In common SVM classification problems, W_i are set to be the same value for all instances, and thus make the hyperplane maximize the margin and average prediction accuracy. However, for this particular problem of prediction block partition early termination, the misclassification of different instances leads to different RD cost increase for two reasons and should not have the same weight.

First, misclassifying positive samples (non-partition blocks) as negative samples (partition blocks) will not cause any RD cost increase, since there is no early termination as Figure 2 shows. Thus, we set $W_i = 1$ for all positive samples $i = N_0+1, N_0+2, ..., N_0+N_1$.

Second reason is that misclassifying different negative samples gives different RD cost increase. Thus, to design the weight for instances of partition mode, we extract two additional features for these training blocks. They are the RD cost of non-partition mode $E_{n,i}$, and the best RD cost $E_{b,i}$ (If non-partition mode gives the lowest RD cost, these two features are equal to the other), where i is the index of the training block. Because misclassifying a block that should be further decomposed as a non-partition block will increase the RD cost by $E_{n,i} - E_{b,i}$, we design the weights for partition blocks as in (3).

$$W_i = C\frac{(E_{n,i}-E_{b,i})}{\frac{1}{N_{p,Vi}}\sum_{j\in V_i}(E_{n,j}-E_{b,j})} \tag{3}$$

where C is a factor controlling the overall weight balance between positive samples and negatives samples. If a larger C is used, misclassifying a negative samples will give a larger cost. V_i is the video the i-th block belongs to, and N_{p,V_i} is the number of blocks with partition modes in V_i. The basic idea behind (3) is that the weight of a partition block is proportional to the RD cost increase of misclassifying this block, and this RD cost increase is normalized by the mean RD cost increase in the video. The normalization is applied because for different videos or the same video with different target bit rates, the Q value will be different, and the Q value as well as the video content have a large impact to the RD cost. Thus, normalizing the RD cost makes the weight in (3) less dependent on other factors and makes the SVM classifiers more generalized.

Experimental results in the experimental result section also show that with the weighted scheme described in (3), the average RD cost increase is less than using a fixed weight for all the negative instances.

Parameter Determination

In this section, we describe a method to choose the weight parameter C based on a validation dataset.

The overview of this algorithm is shown in Figure 3.

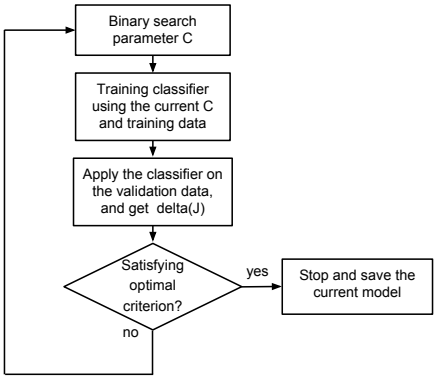


Figure 3: The overview of the optimal parameter selection algorithm.

There are two factors we need to take into account when choosing the parameter C - RD cost caused by misclassification and time saved by early termination. We model the parameter selection process as an optimization problem:

$$\max_C \Delta T$$
$$\text{s.t.} \quad \Delta J < J_T \tag{4}$$

where $\Delta T = N_{termination}/N_{all}$. $N_{termination}$ is the number of blocks that are early terminated by the offline trained classifier on the validation set. N_{all} is the number of all blocks in validation set. Thus, ΔT can reflect the time reduction by our early termination algorithm. ΔJ is defined as:

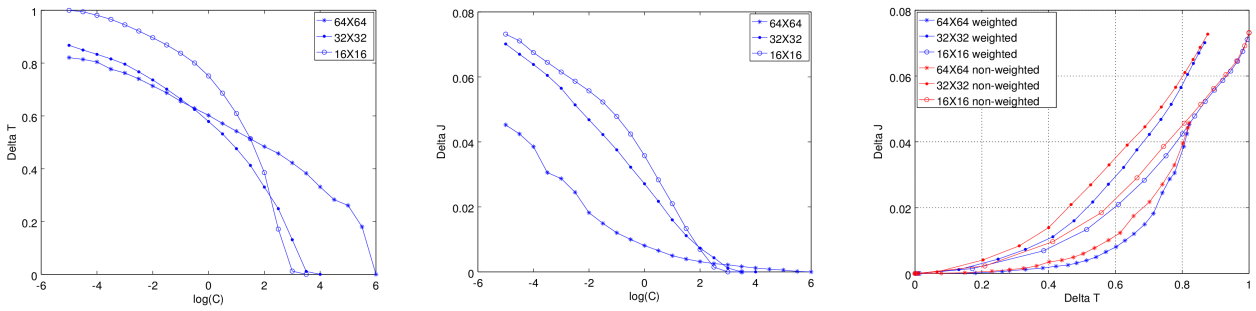
$$\Delta J = \frac{1}{N_{val}} \sum_{V_j \in validation} \left(\frac{\sum_{i \text{ in } V_j \text{ that is misclassified}} (E_{n,i}-E_{b,i})}{\sum_{all\ i \text{ in } V_j} E_{b,i}}\right) \tag{5}$$

where N_{val} is number of validation videos. Thus, ΔJ is the average normalized RD cost increase. Larger ΔJ means larger RD cost increase, and will bring larger performance drop. J_T is a threshold setting by users, which controls the trade-off between the computational complexity reduction and the performance drop. If J_T is large, large RD cost degradation is allowed, so we can early terminate more blocks and save more encoding time.

By solving this optimization problem, we can choose the optimal parameter C that maximizes the time reduction while not causing too much RD cost increase.

Figure 4 shows the relationships of ΔT, ΔJ and C for the validation data. Figure 4 (a) and (b) show how ΔT and ΔJ change w.r.t. C in a log scale. When C increases, the penalty of misclassifying partition blocks (negative samples) also increases. As a result, less negatives samples are misclassified in order to minimize the objective function in (2) which leads to smaller ΔT and ΔJ.

DOI: 10.2352/ISSN.2470-1173.2016.2.VIPC-236



(a) (b) (c)

Figure 4: (a) ΔT decreases as C increases. (b) ΔJ decreases as C increases. (c) The relationship betweenΔJ and ΔT for weighted and non-weighted SVMs on validation data. The weighted SVMs we use can achieve the same ΔT only sacrificing less ΔJ.

Plus, in Figure 4(c), we also show the relationship between ΔT and ΔJ for both our weighted SVM and non-weighted SVM (In this paper, we define the non-weighted SVM as the SVM whose negative weights are equal and positive weights are equal as well. In this case, $W_i = C$ for all negative instances). From Figure 4(c), we can observe that the proposed weighted SVM can achieve higher time saving while sacrificing lower RD loss increase. For example, for 32×32 blocks, if we set J_T to 0.02, non-weighted SVM can achieve 0.458 ΔT, while weighted SVM gives 0.515 ΔT. So this weighted SVM can save $\sim$12.5% more encoding time with same RD loss increase), which proves the effectiveness of our algorithm. We also show the supreme of our algorithm over non-weighted SVM in the experimental result section on some testing videos.

Obtaining the approximate optimal solution of this optimization problem is not difficult, since ΔT is approximately proportional to ΔJ and inversely proportional to C as Figure 4 shows. Binary search is applied to find optimal C that satisfies (4) in a log scale $-10 < \log(C) < 10$.

Experimental Results

We use the publicly available implementation of linear SVM [7] to train the linear classifier. The dataset used to evaluate our method is the standard HD (STDHD) dataset. For the videos in this dataset, we choose four videos with different encoding difficulties: blue_sky_1080p25, city, crowd_run_1080p50, old_town_cross_420_720p50. These videos are encoded with 10-14 different target bit rates (These bit rates are chosen such that the Q values of the encoded videos can cover the range of possible Q values). The clips that are used for training are the 2-20 frames of these videos. Here we skip the first two frames because the first frame is a key frame that does not belong to INTER frame, and the last context of the second frame is obtained in the first frame (a key frame) which is not very reliable. 21-100 frames of these four videos are used as validation dataset for choosing the parameter C.

For the testing, time saving T_{saving} is defined as in Equation (6) to represent the encoding time reduction.

$$T_{saving} = \underset{\text{all testing videos}}{mean} \frac{T - T_{term}}{T} \tag{6}$$

where T is the original encoding time in VP9 encoder, and T_{term} is the encoding time of our early termination algorithm. The average time saving of all training videos is T_{saving}.

Bjøntegaard delta bit rate (BDBR) [8] is used to evaluate the quality of the encoding.

Comparison between the proposed method and speed 1 option in VP9

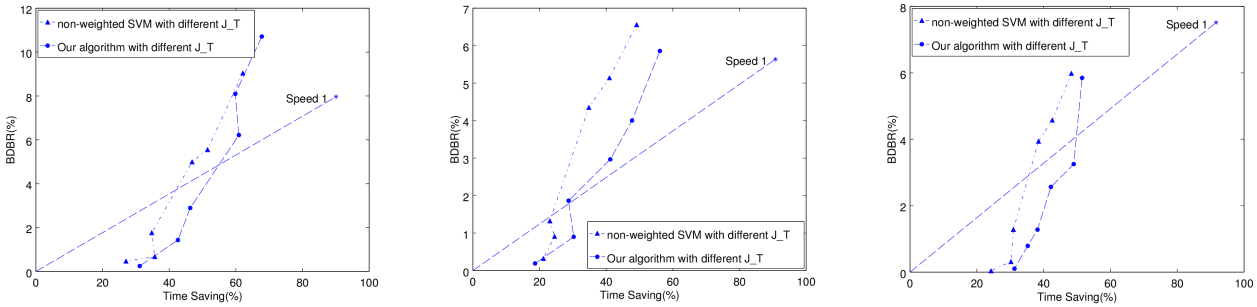
For VP9 encoder, many other encoding options are used to speed up the original speed 0 settings. For example, VP9 provides some speed settings that users can use to turn off some operations to make the encoder faster. It is very important to compare our method with these speed settings because it would be meaningless to use this method if it is much worse than other speed settings. Since speed 1 in VP9 can already achieve a T_{saving} of $\sim$90%, so we compare our method by training on speed 0 settings with six different J_T (0.01%, 0.005%, 0.1%, 0.2%, 0.3%, 0.5%). Here we choose the same J_T for different block sizes, but we may also choose different J_T for different block sizes as in [4] and expect better results. After training, these classifiers are tested on all the videos in STDHD dataset. The relationship between T_{saving} and BDBR of speed 1 and our algorithm with different J_T are shown for some selected videos in Figure 5.

As Figure 5 shows, the weighted SVMs can give less quality drop than non-weighted ones, which means the weights we design for the weighted SVM are effective for the early termination problem. When J_T is low, the algorithm can achieve a big time saving with neglectable quality loss. However, when J_T becomes larger, the BDBR increases in an exponential trend. In these figures, we also draw a line between origin (speed 0) and speed 1. The slope of this line indicates how much the quality drops for saving a unit time. Thus, if one method is better than speed 1, the point corresponding to this method should be under the line from original to speed 1. As we can see in these figures our method is better than speed 1 when the J_T is not very large.

In Table 1, we show the time saving and BDBR for all videos in STDHD dataset for three values of J_T. This table tells us that the proposed method can give 15% less encoding time with less than 0.1% BDBR, 29% less encoding time with 1.2% BDBR, and 48% less encoding time with less than 3% BDBR.

Table 2 shows the corresponding number of blocks skipped and time saving of our method for the video Crew when $J_T = 0.5\%$. By observing this table, we can find that as the target bit rate increases the number of skipped blocks decreases as well as the time saving. This is because when the target bit rate is high,

©2016 Society for Imaging Science and Technology
DOI: 10.2352/ISSN.2470-1173.2016.2.VIPC-236



(a) Jets (b) Cyclist (c) Crew

Figure 5: Our algorithm and non-weighted SVM trained and tested on speed 0 settings compared with speed 1.

we have more bits to encode the blocks, so we are able to split the blocks into smaller ones to reduce the RD cost. Thus, early termination will not frequently happen in this case.

Table 1: Performance for all videos in STDHD dataset for selected J_T value.

	$J_T = 0.01\%$		$J_T = 0.1\%$		$J_T = 0.3\%$	
videos	BDBR (%)	T_{saving} (%)	BDBR (%)	T_{saving} (%)	BDBR (%)	T_{saving} (%)
old town	0.1278	24.68	0.5570	43.68	3.9578	65.98
city	0.2307	8.49	2.0705	31.66	5.3990	58.60
blue sky	0.0479	12.51	0.3105	25.04	1.5914	46.38
crow run	-0.0060	7.49	0.0398	15.04	-0.3100	29.31
shields	0.1053	19.17	1.1375	36.73	2.3617	56.23
mobcal	0.0731	20.29	1.5483	42.42	2.8988	68.02
cyclists	0.1900	18.67	1.8642	28.71	4.0032	47.76
night	0.2355	10.71	0.6341	26.31	2.0268	43.50
sunflower	0.4593	18.16	2.1567	53.89	3.6920	65.93
jets	0.2474	31.15	2.8954	46.32	8.0961	59.83
ped	0.1333	4.23	2.1786	23.57	4.5845	47.65
sheriff	-0.3029	4.50	1.2014	16.67	2.5842	35.64
crew	0.1031	31.26	1.2791	38.17	3.2529	49.00
riverbed	0.0024	16.75	0.0456	11.44	0.3550	19.69
park joy	0.0291	-2.37	0.1164	0.45	0.0642	23.78
average	0.0947	15.05	1.2023	29.34	2.9705	47.82

Table 2: Number of blocks skipped and time saving for video Crew by our method when $J_T = 0.5\%$.

Target bit rate (kbps)	skipped 64X64 blocks	skipped 32X32 blocks	skipped 16X16 blocks	Time Saving (%)
100	22942	2317	14185	69.47
200	23325	3314	19169	69.74
400	22639	5299	33726	66.61
600	21241	5976	44249	62.08
800	20697	5775	47954	59.59
1200	19050	5400	46030	65.73
1600	18136	5201	41800	52.53
2000	16761	5144	37808	50.76
4000	11067	6193	36155	57.88
6000	6731	4726	32859	43.63
8000	5547	4135	24321	37.10
10000	4630	3257	19489	27.72
12000	4182	2238	15086	6.873

Comparison between the proposed method and speed 2-3 option in VP9

We also train our classifiers on the speed 1 setting and compare the results with the speed 2-3 settings in the sense of BDBR vs time saving w.r.t. speed 1, and the results are in Figure 6.

From Figure 6, we can draw the similar conclusion as in Figure 5 - Our algorithm can achieve relatively less BDBR with the same time saving compared with the other speed settings in VP9 encoder.

Conclusion and Future Work

This paper proposes a method for early terminating the block partitioning process in VP9 encoder. This method first extracts rich features of the non-partition mode, and uses these features to train a linear weighted SVM for determining whether it should be early terminated or not. To train a classifier that can reduce the encoding time without too much quality drop, we design a set of normalized weights for the training data based on the RD cost increase by misclassification. Further, an optimal parameter selection method based on the validation dataset is introduced to find the best parameter for the SVM. Experimental results on the STDHD (720p and 1080p videos) dataset show that the proposed method is effective.

Some future directions of improving this algorithm include speeding up normalizations and multiplications during test time; applying different J_T for different block sizes; utilizing more complicated machine learning techniques, such as kernel SVMs, neural network, etc; exploring more features; using online training method to adaptively adjust the offline trained classifier based on the current testing video.

References

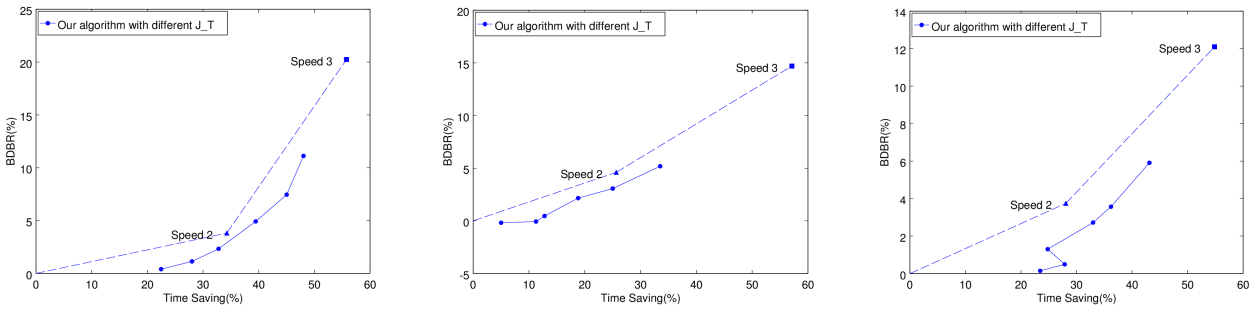
[1] Debargha Mukherjee, Jim Bankoski, Adrian Grange, Jingning Han, John Koleszar, Paul Wilkins, Yaowu Xu, and Ronald Bultje. "The latest open-source video codec VP9-an overview and preliminary results," *IEEE Picture Coding Symposium (PCS)*, pp. 390-393, 2013.

[2] Xiaolin Shen, and Lu Yu, "CU splitting early termination based on weighted SVM," *EURASIP Journal on Image and Video Processing*, no. 1, pp. 1-11, 2013.

[3] Jongho Kim, Seyoon Jeong, Sukhee Cho, and Jin Soo Choi, "Adaptive coding unit early termination algorithm for HEVC," *IEEE International Conference on Consumer Electronics (ICCE)*, pp. 261-262, 2012.

[4] Yun Zhang, Sam Kwong, Xu Wang, Hui Yuan, Zhaoqing Pan, and

DOI: 10.2352/ISSN.2470-1173.2016.2.VIPC-236



(a) Jets (b) Cyclist (c) Crew

Figure 6: Our algorithm trained and tested on speed 1 settings compared with speed 2 and 3.

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.

[5] Liquan Shen, Zhi Liu, Xinpeng Zhang, Wenqiang Zhao, and Zhaoyang Zhang, "An effective CU size decision method for HEVC encoders," *IEEE Transactions on Multimedia*, vol. 15, no. 2, pp. 465-470, 2013.

[6] Ryeong-hee Gweon, and L. E. E. Yung-Lyul, "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.

[7] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin, "LIBLINEAR: A library for large linear classification," *The Journal of Machine Learning Research*, vol. 9, pp. 1871-1874, 2008.

[8] Gisle Bjontegaard, "Improvements of the BD-PSNR model." ITU-T SG16 Q ,pp. 35, 2008.

Author Biography

Xintong Han received his BS in Electrical Engineering from Shanghai Jiao Tong University, Shanghai, China (2013), and he is now pursuing his PhD at the University of Maryland, College Park, MD, USA. He was an software engineering intern at Google in the summer of 2015. His research interests including computer vision, machine learning and multimedia.

Yunqing Wang is a software engineer at Google, and has been working on VPx series video codec development since 2007. Dr. Wang received the B.S. and Ph.D. degrees in Precision Instrument from Tsinghua University in 1991 and 1996, respectively, and the M.S. degree in Computer Science from University of Massachusetts at Amherst in 2002. Her research interests include video compression, optimization, and machine learning.

Yaowu Xu is currently the Tech Lead Manager of the video coding research team at Google. The team has been responsible for developing and defining VP9, the core video technology of the WebM project. Prior to joining Google, Dr. Xu was the Vice President of Codec Development at On2 Technologies. He was the co-creator of On2's VPx series codecs including VP32, VP4, VP5, VP6, VP7 and VP8. These codecs were broadly adopted by the industry and have fueled the phenomenal growth of web video. Dr. Yaowu Xu's education background includes the BS degree in Physics, the MS and PhD degree in Nuclear Engineering from Tsinghua University at Beijing, China. He also holds the MS and PhD degree in Electrical and Computer Engineering from University of Rochester. Dr. Xu has published many technical papers in the area of image processing on leading journals and international conferences. He also holds many patents and has numerous patent applications pending in the area of digital video compression. Dr. Xu's research and development experiences include digital video compression and processing, real time video encoding and decoding, mobile video, image processing, pattern recognition and machine learning. His current research focuses on advanced algorithms for digital video compression.

Jim Bankoski is an Engineering Director working on Google's WebM project. He's the former CTO of On2 Technologies, and a technical contributor to all of On2's and later Google's codecs from Tm2x through VP9; including video codecs widely used in Flash and Skype and now WebM.