TIME-EFFICIENT LEARNING THEORETIC ALGORITHMS FOR H.264 MODE SELECTION

Ashish Jagmohan

Beckman Institute
University of Illinois, Urbana-Champaign, USA

Krishna Ratakonda

IBM T. J. Watson Research Center
Yorktown Heights, USA

ABSTRACT

The H.264 video coding standard derives much of its compression efficiency gain from the use of multiple different macroblock prediction modes for macroblock coding. In general, finding the prediction mode which gives optimal R-D performance for a given macroblock requires the encoder to completely encode the macroblock using all possible prediction modes. This results in a significant increase in encoder computational complexity. In this paper, we present a mode selection framework for H.264 which uses learning theoretic classification algorithms to discern between broad mode classes, based on the evaluation of a simple set of macroblock features. We show that the proposed mode selection framework significantly reduces encoder computational complexity, at the cost of only a small loss in compression performance.

1. INTRODUCTION

The H.264 standard [1] provides state-of-the-art video coding with compression gains of up to 50% over previous standards such as H.263 and MPEG-4. The key features responsible for this increased efficiency are: (1) The use of an efficient context-based arithmetic coding (CABAC) algorithm for entropy coding, (2) The use of quarter-pixel accurate motion vectors, and (3) The use of multiple macroblock prediction modes and block sizes for intra- and inter-prediction of macroblocks. However, this increased efficiency comes at the cost of a large increase in the encoder computational complexity. While algorithms for efficient CABAC encoding and fast motion search are well known, the highly complex "mode space" of the H.264 standard makes the design of a fast mode selection algorithm particularly challenging. In this paper, we will present such an algorithm for macroblock mode selection given a fixed macroblock quantization scale, and compare its performance to the reference encoder provided by the baseline profile of the standard.

Macroblock mode selection in H.264 is based on the use of rate-distortion (R-D) optimization algorithms. These algorithms formulate the mode selection problem as one of minimizing the bit-rate of the compressed stream, for a given distortion constraint [2]. More precisely, given a set of possible prediction modes $\{C_k\}$ and a quantization scale for the current macroblock, R-D optimization of the mode selection process involves selecting the prediction mode c_{opt}, which minimizes the following Lagrangian functional:

$$c_{opt} = \arg \min_{c \in \{C_k\}} D(c) + \lambda R(c) \quad (1)$$

where λ is a Lagrange multiplier, R is the bit-rate required to encode the macroblock using mode c, and D is the quantization distortion incurred (measured as the mean-square error distortion).

The bit-rate required by CABAC for a given prediction mode is difficult to estimate a-priori, without actually performing CABAC coding using the mode. Thus, to find the prediction mode which minimizes the cost function given in (1), an H.264 encoder is forced to completely encode the current macroblock using all possible prediction modes. While this optimization significantly improves coding efficiency, it also significantly increases the encoder's computational complexity. As an example, the reference JM 6.1 encoder [3] takes up to three seconds to encode one inter coded frame of a CIF sized (352×288) video sequence on a 2 GHz Pentium-4 processor.

The encoder's computational complexity may be significantly reduced by the use of well-designed approximations to the cost function in (1). The reference encoder provides one such approximation, termed the SATD cost function, which uses the sum of absolute values of a 2-D Hadamard transform of the prediction error, for mode selection. While this reduces the computational complexity at the cost of minimal compression performance loss, minimizing the SATD cost function still requires a search over all possible prediction modes. An alternate approach for macroblock 4×4 intra mode selection is proposed in [4], which uses efficient search algorithms to reduce the complexity of evaluating the SATD cost function. The key shortcoming of this approach is that it does not provide mode selection methodologies to select between intra 4×4, intra 16×16 and inter prediction modes.

In the present paper, we propose an H.264 mode selection framework which uses learning-theoretic classification algorithms to discern between broad mode classes (such as intra and inter modes), based on the evaluation of a set of simple macroblock features. Time-efficient search algorithms are subsequently used to further select a specific prediction mode from the chosen mode class. Computational complexity is reduced since only the modes from the chosen mode class need to be considered by the search algorithms. Results indicate that the proposed algorithms significantly reduce encoder computational complexity while causing a loss of only about 0.5-0.7 dB in reconstruction PSNR at the same bit-rate as the reference implementation.

2. PROPOSED MODE SELECTION FRAMEWORK

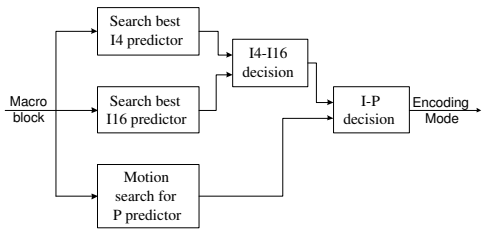

Fig. 1. The mode selection methodology employed by the reference H.264 encoder.

The H.264 standard provides three main classes of prediction modes for luma macroblocks. The key difference between the three classes is the manner in which the predictor for the current macroblock is generated. The Intra16×16 (I16) prediction modes generate a 16×16 predictor for the current macroblock using previously encoded neighboring macroblocks from the current frame.

0-7803-8554-3/04/$20.00 ©2004 IEEE.

The Intra4×4 (I4) prediction modes generate 4×4 predictors for each of the 16 constituent 4×4 blocks of the current macroblock. The 4×4 predictors are generated using previously encoded neighboring blocks from the current frame. Note that this interaction between neighboring 4×4 blocks within a 16×16 macroblock makes it particularly difficult to deveop an approximate search algorithm. The H.264 standard further provides several prediction modes within the two intra mode classes—specifically it provides nine I4 prediction modes and four I16 prediction modes. Finally, the Inter (P) prediction modes generate a predictor for the current macroblock based on data from prior frames. This involves the use of motion search algorithms for finding the motion vector which produces the best motion compensated predictor for the current macroblock. In this paper we only consider the inter modes available in the baseline profile of the standard (which does not have bi-directional or B-pictures).

Inter prediction in H.264 is complicated by the presence of several different block sizes for motion compensation, such as 8×16, 16×8, 8×8 etc. Since our primary goal is to test our mode selection algorithm and not to obtain an optimal fast motion search technique, we will only consider the inter prediction mode with 16×16 block size, obtained through a variant of diamond search algorithm in this paper. The reference implementation is restricted to use only 16×16 block size, obtained using an exhaustive search.

Chroma macroblocks may be coded using intra or inter prediction modes. However, the luma mode class selection implicitly enforces the use of a specific chroma mode class and the individual chroma modes are direct analogues of the luma modes. Thus, for the remainder of this paper we will restrict discussion to mode selection for luma macroblocks, with the described techniques being equally applicable to chroma macroblocks.

To sumarize, a mode selection algorithm involves choosing between the I16, I4 and P prediction mode classes. Furthermore, if I16 or I4 prediction is selected, mode selection involves finding the best prediction mode within the chosen intra mode class. If the P prediction mode is to be used, the best motion compensated macroblock predictor is required to be found. The efficacy of a particular prediction mode is measured by the cost it induces, either for the Lagrangian functional given by (1), or by an approximation such as the SATD cost described in Section 1.

The H.264 mode selection algorithm, as implemented by the reference encoder, is shown schematically in Fig. 1. As shown, the H.264 mode selection algorithm can be briefly described as follows. The encoder selects the best I4 prediction mode and the best I16 prediction mode for coding the current macroblock. These two prediction modes are compared to each other using one of the aforementioned cost functions, and the less efficient predictor is discarded. The encoder uses motion compensation to find the best P mode predictor, which is then compared to the best intra mode predictor found. The more efficient predictor of these two is then used for encoding the current macroblock.

The main shortcoming of the H.264 mode selection algorithm is that all possible predictor modes are evaluated during mode selection. Each such mode evaluation is quite costly in terms of computational complexity—for example, a mode evaluation using the R-D optimized cost function requires the macroblock to be differentially encoded with respect to the generated predictor, with the error being transform coded, quantized and CABAC coded. The key concept underlying the proposed H.264 mode selection framework is that if we can infer the right mode class using simple heuristics, only the modes belonging to the selected class need be evaluated by encoding the macroblock. To this end, we propose the use of learning-theoretic approaches that operate on a set of simple macroblock features to select between the three main mode classes.

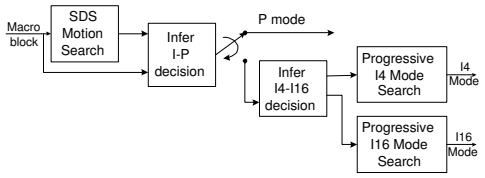

Fig. 2. The proposed mode selection methodology.

Fig. 2 shows a schematic of the proposed mode selection methodology which may be briefly described as follows. A fast motion compensation algorithm, termed the stabilized diamond search (SDS) algorithm, is used to find the best motion compensated P mode predictor. A learning-theoretic classification algorithm, which operates on a set of transform domain features extracted from the macroblock and the P mode predictor, is used to select between the P mode class and the intra mode (I16, I4) classes. This classification algorithm does not require the evaluation of either the SATD or the R-D cost functions. If the P mode is selected the algorithm terminates. If intra prediction is selected, a second classification algorithm is used to select between the I4 and the I16 prediction mode classes. Finally, once the correct mode class has been inferred, a specific mode belonging to the chosen mode class is selected by a time-efficient search algorithm, which finds an approximate solution to the SATD cost function through a progressive search with early termination.

The proposed algorithm reduces the encoder computational complexity in two principle ways. Firstly, the use of learning-theoretic classification for mode class selection implies that the SATD cost function needs to be evaluated for only a subset of prediction modes. Secondly, the progressive search algorithms reduce the computational complexity of evaluating the SATD cost function for each required mode, while the SDS motion compensation algorithm reduces the complexity of the motion search.

3. MODE SELECTION DETAILS

In this section we present details of the proposed macroblock prediction mode selection framework.

3.1. I16-I4 Mode Class Selection

For a given macroblock, the selection between I16 and I4 modes is made by employing supervised binary classification [5] using a set of transform domain macroblock features.

Consider a data set $\{x_i\}_{i=1}^{n}$, such that each element x_i belongs to one of two classes C_1,C_2. The aim of binary classification is to infer the class to which each x_i belongs, on the basis of a set of extracted features $\{f_j(x_i)\}_{j=1}^{m}$. Supervised binary classification does this using two phases—a training phase and a test phase. During the training phase, a set of training data $\{y_k\}_{k=1}^{K}$ with known class membership labels (C_1 or C_2) is presented to the classifier, which uses this data and the class labels to learn a classification function $\Lambda(\cdot)$, such that the classification rule

$$y_k \in C_1 \qquad \Lambda(\{f_j(y_k)\}_j) = 1$$
$$y_k \in C_2 \qquad \Lambda(\{f_j(y_k)\}_j) = 0$$

minimizes the misclassification rate on $\{y_k\}$. The form of the classification function Λ is constrained by the type of classifier used—for example, linear discriminant analysis yields classifier functions which are linear in the feature space. During the test phase, the learned classification function is used to classify the unknown data $\{x_i\}$. The key issues in supervised binary classification are the choice of an appropriate classifier type, and the extraction of a set of features which allows good discrimination between the two classes.

For I16/I4 discrimination, the 16×16 current macroblock m was downsampled to a 4×4 array, denoted m_4. Denoting the 4×4 Hadamard transform of m_4 as $\mathbf{T}_H m_4$, the following set of features was found to be effective: (1) The macroblock high frequency content $f_1 = \sum_{i=2}^{4} \sum_{j=2}^{4} |\mathbf{T}_H m_4(i,j)|$, (2) The macroblock horizontal frequency content $f_2 = \sum_{j=2}^{4} |\mathbf{T}_H m_4(0,j)|$, and (3) The vertical frequency content $f_3 = \sum_{i=2}^{4} |\mathbf{T}_H m_4(i,0)|$. Intuitively, the I16 mode performs better when f_1, f_2 and f_3 are low for the macroblock to be encoded. For the classifier type, we examined several classifiers, including linear and quadratic Gaussian classifiers [5] , the Adaboost classifier [6], and classifier trees [7]. Classifier trees were found to yield the best classification performance.

During the training phase, macroblocks from a large number of video sequences were used as a training set, and the I16-I4 mode decisions selected by the R-D optimized reference H.264 encoder were used as the known class labels. The classifier tree was trained on this data using the tree-partitioning procedure described in [7]. The training was performed at multiple quantization scales. While the structure of the classifier tree was found to remain fixed, the decision thresholds were found to vary with changing quantization scale. Polynomial interpolation functions were used to compute the decision tree thresholds as a function of the encoder quantization scale.

For encoding a macroblock from a new video sequence (i.e. a sequence not in the training set), the feature set $\{f_j\}_{j=1}^{3}$ described above is extracted and the learned classifier tree is used to infer the I16-I4 decision. Since the learning is performed completely off-line, the proposed mode selection methodology requires minimal encoding complexity for making the I16-I4 selection. The SATD cost function is then required to be evaluated for only the selected mode class, as opposed to the case in conventional H.264 encoding.

3.2. Progressive Intra Mode Search

Once the I16-I4 selection has been made as described in Section 3.1, the modes belonging to the selected mode class need to be evaluated using the SATD cost function, for selection of the best intra prediction mode. We illustrate the proposed mode selection procedure for the I4 mode class. This involves selecting the prediction mode for each 4×4 block of the current macroblock, from the nine available I4 modes.

Denote the current 4×4 block to be coded as b, denote the prediction mode to be evaluated as $c_{k,k\in\{1,\ldots,9\}}$, and denote the 4×4 predictor generated by using c_k as p_k. The SATD cost function for the prediction mode c_k is given by

$$\sum_{i=1}^{4} \sum_{j=1}^{4} |\mathbf{T}_H(b - p_k)(i,j)| + \lambda R_h \qquad (2)$$

where λ is a known Lagrange multiplier, and R_h is a fixed penalty which is 0 for the *most probable mode*,[1] and is 1 for all other modes. The reference H.264 encoder evaluates the cost function in (2) completely for all candidate modes, to find the best mode.

We propose an alternative progressive search algorithm, with early termination, to reduce the computational complexity of this search. The proposed algorithm divides the pixels of the 4×4 block into four sets $s_{l,l=1,\ldots,4}$ of four pixels each. The SATD cost for the most probable mode is computed using all sixteen pixels, and is denoted e_{mpm}. The evaluation of the other modes proceeds in four steps. Denoting the 2×2 Hadamard transform as $\mathbf{T}_{H_2}$, at step l $(1 \leq l \leq 4)$, the following operations are performed:

- For each prediction mode c_k, computation of the per-set cost for the pixel set s_l, denoted e_{lk}, computed as $e_{lk} = \sum_{|s_l|} |\mathbf{T}_{H_2}(b(s_l) - p(s_l))|$
- For each prediction mode c_k, computation of the cumulative cost, denoted e_{lk}^c, computed as $e_{lk}^c = \sum_{i=1}^{l} e_{ik}$
- Termination of mode c_k if $|e_{lk}^c - \frac{l}{4} e_{mpm}| \geq \tau_l$, where $\{\tau_l\}_{l=1}^{4}$ are precomputed thresholds.

The algorithm terminates when only one mode is left or when all four steps are completed at which point the surviving mode with the lowest SATD cost is selected for encoding.

In practice we found that a significant proportion of modes were terminated early. On average, the progressive search algorithm reduced the mode evaluation computation complexity by more than half, with a negligible effect on compression performance. Finally, we note that the I16 mode search is performed analogously to the I4 mode search, by dividing the 16×16 macroblock into sets of sixteen pixels.

3.3. I-P Mode Selection

We use a fast motion search algorithm, termed the Stabilized Diamond Search (SDS) algorithm, for generating the motion compensated P mode prediction for the current macroblock. The SDS algorithm is akin to the diamond search algorithm proposed in [8], with the main differences being the use of multiple start locations to improve motion compensation performance, and the use of progressive error evaluation with early elimination to reduce computational complexity. The SDS algorithm was found to be significantly faster than the exhaustive motion search used by the reference H.264 encoder, with only a small compression performance loss.

The selection between inter and intra prediction is again made through supervised binary classification using decision trees. The features used for classification were as follows: (1) The features f_1, f_2, f_3 described in Section 3.1, and (2) The absolute sum of the transform coefficients of the prediction error between the downsampled current macroblock m_4, and the downsampled motion compensated predictor p_4, i.e. $f_4 = \sum_{i=1}^{4} \sum_{j=1}^{4} |\mathbf{T}_H(m_4 - p_4)(i,j)|$. As in Section 3.1, training was performed by using the R-D optimized I-P modes generated by the reference H.264 coder on a training set of video sequences.

As can be seen in Fig. 2, the selection of the I-P mode based on classification obviates the need for evaluating the I4 and I16 modes when inter prediction is selected by the classifier—this represents a significant reduction in the encoding complexity for P macroblocks. In the case when intra prediction is selected by the classifier, a specific intra prediction mode can be selected using the algorithms outlined in Sections 3.1 and 3.2.

4. RESULTS

The performance of the proposed mode selection method was evaluated using a set of twelve standard 352×240 color test sequences. The test sequences used were distinct from the training sequences used for learning the classifiers. A composite evaluation sequence, termed *eval_seq*, was created by putting together randomly selected subsequences of several frames from the twelve test sequences. The proposed mode selection method, the reference H.264 encoder with R-D optimized mode selection, and the reference encoder with SATD optimized mode selection were used to compress the *eval_seq* sequence at multiple quantization scales.

Fig. 3 shows the performance of the classifiers for inferring the mode class decisions. This was evaluated by comparing the mode class decisions by the R-D optimized mode selection, to that of the

[1] In H.264, the *most probable mode* is a mode which requires fewer header bits (in the case of CAVLC prediction) to encode than the other 4×4 modes. Its value is derived from the modes used to encode previously encoded neighboring blocks.

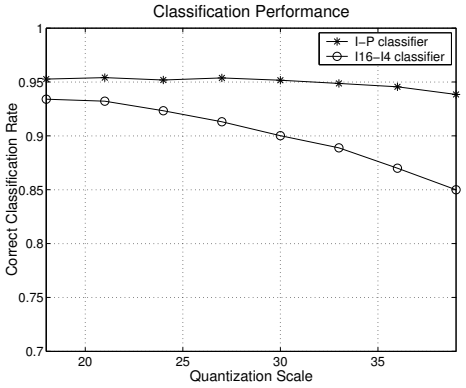

Fig. 3. Rate of correct classification of the I-P and I16-I4 classifiers used in the proposed mode selection.

mode classes selected by the proposed method.[2] As can be seen, the proposed I-P classifier inferred the same decision as the R-D optimized selection for about 95% of all macroblocks. The I16-I4 classifier inferred the R-D optimized decision for 85-95% of the intra macroblocks. The key point to note is that the classifiers inferred these mode decisions using far fewer computations than that required by the R-D optimization. We now see how this impacted the compression and computational performance of the encoder.

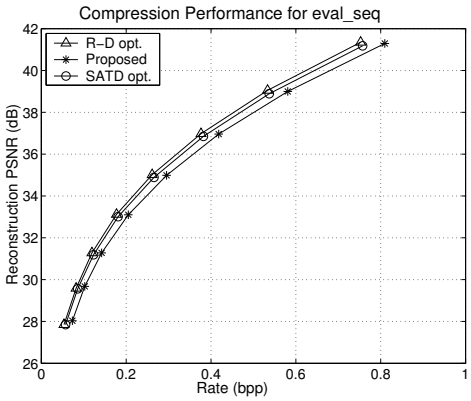

Fig. 4. Compression performance for proposed encoder, R-D optimized reference encoder, and SATD optimized reference encoder.

Fig. 4 compares the compression performance for the *eval_seq* sequence of the reference H.264 encoders employing R-D optimized mode selection and SATD optimized mode selection, to that of an H.264 encoder employing the proposed mode selection methodology (termed the proposed encoder). As can be seen, the proposed encoder provides reconstruction PSNR which is within $0.5 - 0.7$ dB of the SATD and R-D optimized encoders, at the same bit-rate. We note that, of this PSNR gap, about $0.3 - 0.4$ dB was due to the use of SDS motion compensation in the proposed encoder, as compared to the full motion search employed by the reference encoder.

Fig. 5 compares the computational complexity of the mode selection methods used by the proposed encoder and the reference SATD encoder. Computational complexity was measured by summing up the number of addition and multiplication operations required at each step of mode selection, other than the motion search—motion search was excluded since the reference encoder uses a high-

[2]For correct evaluation, this required that all macroblocks encoded before the current macroblock used the same prediction modes for the two mode selection cases—e.g. the prediction modes selected by the R-D cost function.

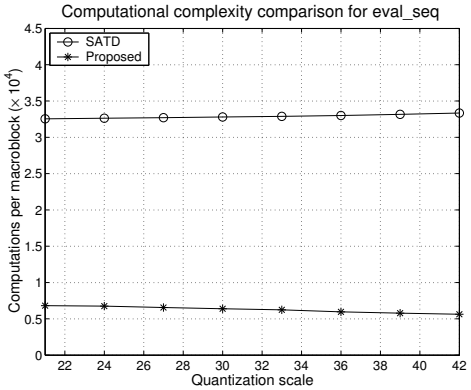

Fig. 5. Computational complexity comparison of proposed encoder with SATD optimized reference encoder.

complexity full search, as compared to the fast SDS search used in the proposed encoder. Fig. 5 plots the number of computations used, per macroblock, in the reference encoder mode selection, and the number of computations used, per macroblock, in the proposed mode selection. As can be seen the SATD mode selection requires 6–7 times more computations than the proposed mode selection. The complexity of the R-D optimized mode selection, which is not shown in Fig. 5, was found to be an average of $3-5$ times higher than that *of the SATD selection*. One reason for this inordinately high complexity is that the R-D optimized selection evaluates all *combinations* of chroma and luma modes for each macroblock.

Fig. 4 and Fig. 5 show that the proposed mode selection yields compression performance close to that of the reference encoder, while requiring significantly reduced computation. In practice, the proposed H.264 encoder was found to encode 352×240 color video sequences at 24-30 frames per second on a 2 GHz Pentium-4 processor. This is an order of magnitude faster than the reference encoder using SATD optimized mode selection, which is itself significantly faster than R-D optimized mode selection.

5. REFERENCES

[1] T. Wiegand, G.J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h.264/avc video coding standard," *IEEE Trans. Ckts. Sys. Vid. Tech.*, vol. 13, no. 7, pp. 560–576, July 2003.

[2] T. Wiegand, M. Lightstone, D. Mukherjee, T.G. Campbell, and S.K. Mitra, "Rate-distortion optimized mode selection for very low bit rate video coding and the emerging h.263 standard," *IEEE Trans. Ckts. Sys. Vid. Tech.*, vol. 6, no. 2, pp. 182–190, Apr. 1996.

[3] http://bs.hhi.de/ suehring/tml/, "H.264/avc software coordination," .

[4] B. Meng, O.C. Au, C. Wong, and H. Lam;, "Efficient intra-prediction mode selection for 4x4 blocks in h.264," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2003, vol. 3, pp. 521–524.

[5] R.O. Duda, P.E. Hart, and D.G. Stork, *Pattern Classification*, Wiley, 2001.

[6] Y. Freund and R. Schapire, "A short introduction to boosting," 1999.

[7] P. Chou, "Optimal partitioning for classification and regression trees," *IEEE Trans. Pat. Anal. Mach. Intel.*, vol. 13, no. 4, pp. 340–354, 1991.

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