IMPROVEMENTS IN EXTENDED PARTIAL DISTORTION SEARCH AND PARTIAL DISTORTION SEARCH ALGORITHMS VQ SEARCH

J. S. Pan†‡, F. R. McInnes†and M. A. Jack†

†Centre for Communication Interface Research, University of Edinburgh, UK
‡Department of Electronic Engineering, Kaohsiung Institute of Technology, Taiwan

ABSTRACT - A new approach for the extended partial distortion search (EPDS) algorithm is proposed to optimize the EPDS algorithm based on the cost ratio of sorting time to dimension-distortion computation time. The result of this approach is the same as using dynamic programming (DP) to improve the computation time in the EPDS algorithm. The partial distortion search (PDS) algorithm can also be improved by determining which dimension is suitable to start inserting comparisons for every codeword. Experimental results confirm the improved computational performance of these new approaches.

INTRODUCTION

Vector quantization (VQ) (Gray, 1984) has widely been used in speech coding, speech recognition, and image coding. One of the most serious problems in VQ is the search cost to find a minimum distortion codeword from a given codebook. In order to reduce the search cost, the partial distortion search (PDS) algorithm (Bei & Gray, 1985) has been proposed. The PDS is one of the simplest methods for fast search. The efficiency of PDS derives from elimination of an unfinished distortion computation if its partial accumulated distortion is larger than the current minimum distortion.

The extended partial distortion search (EPDS) algorithm (Chen & Pan, 1989) is a modified version of PDS which optimizes the calculation in terms of the number of multiplications for a minor overhead in data sorting. It can be used in vector encoding and word recognition. For the vector encoding, it computes the dimension-distortion for the first dimension of the input vector to the first dimension of all codewords, then sorts the dimension-distortion to obtain the nearest codeword. The distortion for the input vector to the nearest codeword in the second dimension is calculated and added to the previous distortion of the same codeword. It sorts the dimension-distortions again to obtain the nearest codeword. The procedure continues until the last dimension-distortion is calculated and the distortion is smallest.

The EPDS algorithm is suitable for computer architectures in which the complexity of comparisons is negligible with respect to that of the multiplications. However, EPDS is less suited to processor architectures such as those based on the Harvard architecture in which comparisons are computationally expensive. An improvement of the extended partial distortion search approach is proposed here. It involves inserting the sorting operation from a suitable dimension to minimize the EPDS search cost for any computer architecture. The PDS algorithm has the same problem as the EPDS algorithm when operated in processors in which the cost of comparison is significant. For the improved PDS method, determining which dimension is suitable to start inserting comparisons for every codeword can be assessed from the training data.

This next section of this paper describes the EPDS algorithm, derives the improved EPDS algorithm and proves theoretically that by using dynamic programming (DP) to decide which dimension is suitable to insert, the result is the same as this approach. It also describes the improvement of the PDS. Database and experimental results are discussed in the following section. Conclusions are given in the last section.

IMPROVED VQ SEARCH ALGORITHMS

Extended partial distortion search

The EPDS is an optimal PDS in the sense of reducing the number of multiplications. It can be used in vector encoding and the frame-distortion accumulation in a word recognition system. The detailed algorithm of the EPDS in vector encoding is described as follows.

step 1: Let $l_i = 1$ and calculate the distortion $D_i = (x_1 - c_{i1})^2$ between the first dimension x_1 of the input vector X and the first dimension c_{i1} of the ith codeword C_i, for $i = 1$ to N. Here N is the number of codewords and l_i is the l_ith dimension for the ith codeword.

step 2: Find $D_s = \mathrm{Min}_i D_i$ and $s = \arg\mathrm{Min}_i D_i$.

step 3: If $l_s = k$, then set the sth codeword to be the best match and terminate the program; otherwise, set $l_s = l_s + 1$, calculate the encoding distortion of x_{l_s} by using the sth codeword and add it to D_s, and go to step 2. Here k is the dimension of the input vector and codewords.

Improved extended partial distortion search

In the improved algorithm, the sorting of the accumulated distortions to find the minimum D_s is performed only after the first j dimensions' distortion terms have been accumulated for every codeword, where j is chosen to minimum the total computation. Let r be the cost ratio of the sorting time to dimension-distortion computation time. To insert the sorting to dimension-distortion accumulation at the jth dimension, the cost of sorting is $m_j r$, but there is a decrease of $N - m_j$ dimension-distortion computations. Here N is the number of codewords and m_j is the average number of codewords whose distortion computation cannot be omitted at the sorting insertion of the jth dimension. From the above description, the following two equations are satisfied.

$$m_j \geq 1, \quad j = 1, \dots, k-1 \tag{1}$$

$$m_j \geq m_{j+1}, \quad j = 1, \dots, k-2 \tag{2}$$

Let $A(j)$ be the global advantage function of inserting the sorting from the jth dimension onwards. The advantage can be expressed in terms of N, k, m_j and r as follows.

$$A(k) = 0 \tag{3}$$

$$A(j) = A(j+1) + V(j), \quad j = 1, \dots, k-1 \tag{4}$$

where $V(j)$ is the local advantage due to sorting at the jth dimension, given by

$$V(j) = (N - m_j) - m_j r = N - (r+1)m_j \tag{5}$$

From equation 2 and 5,

$$V(i) \geq V(j) \;\; \text{if} \;\; i > j \tag{6}$$

Hence there is some t ($1 \leq t \leq k$) such that

$$V(j) \geq 0 \quad \text{for} \quad j = t, \ldots, k-1 \tag{7}$$

$$V(j) \leq 0 \quad \text{for} \quad j = 1, \ldots, t-1 \tag{8}$$

and so

$$A(t) \geq A(j), \quad j = 1, \ldots, k, \quad t \neq j \tag{9}$$

This value t is the optimal sorting insertion dimension for the given value of the cost ratio r.

From equations 2, 4, 5 and 9, the cost interval r_t corresponding to the sorting insertion dimension t can be derived.

$$0 \leq r_t \leq \frac{N}{m_t} - 1, \quad t = 1 \tag{10}$$

$$\frac{N}{m_{t-1}} - 1 \leq r_t \leq \frac{N}{m_t} - 1, \quad t = 2, \ldots, k-1 \tag{11}$$

From the training data, calculate the cost interval r_j, j = 1 to $k-1$. The optimal sorting insertion is from the jth dimension if the cost ratio of the computer architecture lies in the cost interval r_j. For the conventional exhaustive full search method, Nk dimension-distortions are computed corresponding to the computation time of Nk multiplications, $N(2k-1)$ additions, and $(N-1)$ comparisons. One dimension-distortion computation involves approximately the computation time of one multiplication and two additions. The sorting time is $N-1$ comparisons for the basic sorting method. The computation time of EPDS and improved EPDS are $Nk - A(1)$ and $Nk - A(t)$. The performance of EPDS, improved EPDS, and the improvements of the improved EPDS are as follows.

$$\text{EPDS performance} = \frac{Nk - A(1)}{Nk} \tag{12}$$

$$\text{Improved EPDS performance} = \frac{Nk - A(t)}{Nk} \tag{13}$$

$$\text{Improvements} = \frac{A(t) - A(1)}{Nk - A(1)} \tag{14}$$

Dynamic programming in EPDS algorithm

To use the dynamic programming in EPDS, the accumulated advantage and the inserting point of control path can be expressed as follows.

$$Da(p) = Max\{Da(i) + V(p)\}, \quad i = 1, \ldots, p-1 \tag{15}$$

$$j = argMax_i\{Da(i) + V(p)\}, \quad i = 1, \ldots, p-1 \tag{16}$$

where p is from 1 to k, j is the inserting point and $Da(0) = 0$.

The local advantage of the next dimension is larger than the present dimension if the sorting is inserted. From the ith dimension to the last dimension sorting must be inserted if the sorting insertion is from the ith dimension. Due to the monotonous property of equation 6, the improved extended partial distortion search algorithm is the same as using dynamic programming (DP) in extended partial distortion search to decide which dimension is suitable to insert sorting operations to reduce computational complexity, i.e., if the sorting is at the jth dimension, then the sorting should be inserted at the kth dimension if k is larger than j. From the above proof and description, the sorting is from the ith dimension if $V(i-1)$ is smaller than or equal to 0 and $V(i)$ is larger than or equal to 0.

Improved partial distortion search

Let r be the cost ratio of the comparison computation time to dimension-distortion computation time. The improved partial distortion search algorithm can be described as follows.

step 1: Set $l = 1$.

step 2: Set $i = 1$ and $d_{min} = \infty$.

step 3: Calculate the distortion d for the ith codeword to the lth training vector. Compute the saving dimension-distortion number M^i_{jl} and the induced comparison number C^i_{jl} at the insertion from the jth dimension for the ith codeword. Set $d_{min} = Min(d_{min}, d)$.

step 4: If $i < N$, set $i = i + 1$ and go to step 3. Here N is the number of codewords.

step 5: If $l < T$, set $l = l + 1$ and go to step 2; otherwise, set $c^i_j = T^{-1} \sum_{l=1}^{T} C^i_{jl}$ and $m^i_j = T^{-1} \sum_{l=1}^{T} M^i_{jl}$. Here $l = 1$ to T and T is the number of training vectors. The comparison starts from $I(i)$ for the ith codeword if $I(i) = argMax_j(m^i_j - rc^i_j)$.

EXPERIMENTAL RESULTS

The speech databases used in training and test experiments consist of one hundred words recorded from five male speakers separated into three sets. The sampling rate used is 16 kHz and 12-dimensional cepstrum coefficients are computed over 20 ms-wide frames with a 5 ms frame shift. The first data set recorded from two speakers is used to generate the codebook. Cost intervals for the improved EPDS algorithm and the inserting dimension of comparison to every codeword for the improved PDS algorithm are computed from the codebook using the second data set recorded from two other speakers. The third data set recorded from the fifth speaker is used to test the performance of these approaches.

Table 1 illustrates cost intervals of 16 codewords and 128 codewords. From these cost intervals and the cost ratio of sorting time to dimension-distortion computation time for a given computer architecture, the dimension of sorting insertion can be decided. For example, the inserting should be from the third dimension if the cost ratio of the computer architecture is 6 for 128 codewords. The performance comparison of improved EPDS and EPDS is shown in Table 2 and Table 3. These efficiencies can be calculated from equation 12, 13, and 14 by using the maximum cost ratio from Table 1. For 128 codewords, if the cost ratio is 7.55, the performance of EPDS is 67%, but that of improved EPDS will be 50% for inserting from the third dimension, it improves 26%.

The 12-dimensional cepstrum coefficients with variance weighting and 256 codewords are used in the experiment of PDS, improved PDS and dynamic programming in PDS (Fissore et al., 1993). The purpose of variance weighting is to equalize the importance of every cepstrum coefficient. The experimental results are shown in Table 4. The performance is compared with the standard PDS. These results show that the performance of the improved PDS is almost the same as using DP to improve the performance of PDS. General speaking, if the cost ratio of computer architecture is smaller or equal to 1.2, it is better to use the improved PDS than DP in PDS for the cepstrum coefficients with variance weighting.

inserting dimension	cost intervals of 16 codewords	cost intervals of 128 128 codewords
1	[0.00 , 1.56]	[0.00 , 2.03]
2	[1.56 , 3.29]	[2.03 , 4.99]
3	[3.29 , 4.32]	[4.99 , 7.55]
4	[4.32 , 6.12]	[7.55 , 13.4]
5	[6.12 , 7.43]	[13.4 , 19.5]
6	[7.43 , 8.92]	[19.5 , 29.3]
7	[8.92 , 10.1]	[29.3 , 39.0]
8	[10.1 , 11.6]	[39.0 , 52.9]
9	[11.6 , 12.6]	[52.9 , 67.7]
10	[12.6 , 13.4]	[67.7 , 84.2]
11	[13.4 , 14.3]	[84.2 , 104]

Table 1: cost intervals of 16 codewords and 128 codewords

inserting dimension	EPDS performance	improved EPDS performance	improvement
1	41 %	41 %	0 %
2	63 %	58 %	8 %
3	76 %	66 %	13 %
4	99 %	77 %	22 %
5	116 %	84 %	28 %
6	135 %	90 %	34 %
7	150 %	93 %	38 %
8	170 %	97 %	43 %
9	182 %	98 %	46 %
10	192 %	99 %	48 %
11	203 %	100 %	51 %

Table 2: the performance of 16 codewords

inserting dimension	EPDS performance	improved EPDS performance	improvement
1	29 %	29 %	0 %
2	49 %	42 %	15 %
3	67 %	50 %	26 %
4	107 %	60 %	44 %
5	148 %	68 %	54 %
6	215 %	78 %	64 %
7	281 %	84 %	70 %
8	376 %	90 %	76 %
9	478 %	95 %	80 %
10	590 %	98 %	83 %
11	724 %	100 %	86 %

Table 3: the performance of 128 codewords

cost ratio	DP in PDS	improved PDS
0.1	0 %	0.82 %
0.3	2.9 %	4.0 %
0.5	7.9 %	7.9 %
0.7	12.4 %	12.1 %
0.9	16.3 %	16.1 %
1.0	17.3 %	18.1 %
1.2	20.8 %	21.5 %
1.5	27.7 %	26.0 %
2.0	33.8 %	31.7 %
3.0	42.6 %	40.5 %
4.0	48.5 %	47.9 %
5.0	55.1 %	53.1 %

Table 4: the performance of DP in PDS and improved PDS (percentage improvement on standard PDS)

CONCLUSIONS

An improved extended partial distortion search approach is proposed. This improves on the efficiency of EPDS by optimizing the dimension at which partial distortion sorting is inserted according to the cost ratio of sorting time to dimension-distortion computation time. The technique can also be applied to frame-distortion accumulation in a word recognition system. The performance of this approach is the same as using DP to decide which dimension is suitable to insert the sorting operation. The improved partial distortion search algorithm is also presented. Experimental results confirm this new approach.

REFERENCES

Gray, R. M. (1984) Vector Quantization, *IEEE Magazine*, 4–29.

Bei, C. and Gray, R. M. (1985) An Improvement of the Minimum Distortion Encoding Algorithm for Vector Quantization, *IEEE Trans. on Communications*, COM-33(10), 1132–1133.

Chen, S. H. and Pan, J. S. (1989) Fast Search Algorithm for VQ-based Recognition of Isolated Word, *IEE Proceedings-I*, 136(6), 391–396.

Fissore, L., Laface, P., Massafra, P. and Ravera, F. (1993) Analysis and Improvement of the Partial Distance Search Algorithm, *ICASSP 1993*, II-315–II-318.