The Gerrymander Problem in Vector Quantization

Harvey A. Cohen
Computer Science and Computer Engineering
La Trobe University
Melbourne Victoria Australia 3083
Email: H.Cohen@latrobe.edu.au

Abstract: In vector quantization applied to image coding the objective is to determine a set of code vectors for the coding of the population vectors of an image. It is clear that each basis vector should represent about the same number of population vectors, and no population vector should be very badly represented. However, various VQ algorithms have a tendency to produce a gerrymander, in which a few code vectors come to be the representative of many or most population vectors, while at the same time some code vectors represent very few of the population. This situation, analogous to a political gerrymander, arises for ill-chosen initial code vectors. We introduce a scheme, to be applied in conjunction with iterative VQ algorithms, that blocks the development of a gerrymander. This new algorithm involves the replacement of non-representative code vectors, by those population vectors that are most poorly represented. Experimental data on the VQ of gray-scale images using progressive (hard) c-means show that the scheme is most effective, and marginally improved when applied in conjunction with the deliberate duplication of the most popular code vector.

Keywords: Vector quantization, improved algorithm, code vector convergence, image coding.

1 Introduction

In scalar quantization a limited set of special values is used as a lossy compressive encoding for a 1D signal, with the error in the mapping of each signal value contributing to the overall distortion. In vector quantization, a multi-dimension signal such as the image values in fixed size sub-blocks of a digital image, is encoded in terms of a set of code vectors (also called prototype vectors)..
The initial interest in scalar quantization was directed to improved coding of 1D signals, such as sound, and with the emergence of digital image processing similar concerns lead to vector quantization. Scalar quantization based on a least squares criterion was formulated by Lloyd[1] and Max[2], and was extended to vector quantization by Linde, Buzo and Gray [3],[4]. The classic first text on image understanding [5] directed attention to (vector) c-means in image analysis. The concept of c-means was extended by Bezdek [6] and Dunn [7] who introduced fuzzy C-Means. (FCM) Recently fuzzy iterative VQ approaches have been described for image coding [8], [9].

The focus of this paper is on gray-scale image coding, where effectiveness is objectively measured using PSNR. . Vector quantization (VQ) applied to image coding involves the partition of an image into a collection of non-overlapping blocks which we call population vectors. We denote the i'th block of the image by x_i (i =0,1 . . N-1), and the r'th code vector by v_r (r = 0 .. c-1). The objective of VQ is to compute the members of a smaller set of p like-size blocks, called code blocks or code vectors, which are to serve as representative (or prototype) vectors for the entire population. Each vector of the population is to be represented by that code vector that is closest to it: mathematically the representative vector v_r of population vector x_i is that code vector for which the representation error for vector i of the population is least over the set of code vectors

$$\Delta(i) = \text{Min} \{ \| x_i - v_r \|^2 \mid r = 0 .. c-1\}$$

where a rms norm is usually utilised.

The aim of VQ is to determine a set of p code vectors so as to minimise the total error E:

$$E = \sum_{i=0}^{N-1} \Delta(i)$$

There are a number of iterative algorithms for performing vector quantization, which start from some initial set of code vectors, and after each scan through the population vectors that constitute the image, produce an updated set of code vectors. However it is well known that for unfortunate choices of initial vectors various iterative algorithms fail. In the next section we examine closely a proto-typical failure, using histogram data for the population vectors represented by each code vector. Our study demonstrates that as iterative VQ algorithms progress, some code vectors come to represent very few population vectors, while other code vectors represent a very high fraction of the population. In this paper we refer to the problem as the gerrymander

problem, as it has some features of the classical political gerrymander:

In a gerrymander, some code vectors represent too many of the population vectors, some code vectors represent very few, and overall the population is very poorly represented. In Fig 1, we show an example of a gerrymander in the VQ of a Lena image with a large code-book of 256 4x4 blocks, where following an unfortunate choice of initial code-book, the number of utilised vectors fell to 10; and these 10 effective vectors are of almost uniform gray-scale.

The issue has been especially discussed by certain researchers with regard to the need for "careful" selection of an initial set of code vectors. In this paper we demonstrate a new approach to the elimination of gerrymanders that is proposed as an adjunct to any iterative statistical or fuzzy VQ algorithm, where a set of prototype vectors is updated completely after each successive pass through an image. The basis of this approach is that after the VQ pass, an image encoding pass is performed to determine how many times each prototype vector is utilised in the image code. The new algorithm (Algorithm S) involves replacement of the drone -- lazy under-utilised coded vectors, with the most poorly represented (= largest Δ) of the population vectors (= image blocks). For comparison, we also examine the Replication or R Algorithm, recently discussed by Uchiyama and Arbib [10], primarily in the context of colour quantization. This method of resisting a gerrymander is by duplication of the most populous representative vectors. The "new" vectors introduced by Replication replace lazy code vectors that represent very few.

2 VQ Gerrymander Algorithms

2.1 VQ Algorithm

The classic VQ algorithm is HCM that proceeds from an initial set of code vectors, v[i] (i =1..c) , with counters h[i] (i =1 ..c) initialised to zero, an image scan is performed, during which each block is labelled with the index of the closest code vector, and the corresponding counter h[i] incremented. At the end of the scan, a new v[i] is formed that is the mean of all the vectors labelled with index [i], and the h[i] re-initialised. This process is repeated to convergence. In classic HCM code vector update only occurs at the end of a complete pass through the image.

The experiments reported in this paper involve use of a progressive form of HCM, similar to one used by Rankin for colour quantization [11], where there is a progressive update of code vector during the image scan. As the scan progresses, as each population vector x[k] (image block) is labelled with index i, the corresponding h[i] is incremented : h[i] ++. AND the corresponding code vector (the winning code vector) is updated [11] according to:

updated v[i] = v[i] + (x[k] - v[i]) /h[i].

It is important to note that at the end of the VQ scan, the total number of population vectors x[k] that will be represented by v[i] won't be equal to the end-of-pass value of h[i] -- due to the progressive change in all v[i] during the scan. To determine the histogram count hist[i] for the code vectors, requires a histogram scan, identical to the VQ scan, except that the v[i] are NOT updated; the value of h[i] at the end of the histogram scan is hist[i] for each i =1 . . c.

2.2 Counter - gerrymander algorithms

Two algorithms are examined as adjuncts to iterative VQ algorithms, one new, one espoused by others. [10].

2.2.1 Algorithm S (introduced here):

After each complete scan of the image, the new codebook is used to determine the code label for each population vector, the popularity count hist[i] for each code vector, and the error for (best) representation of each population vector. At the end of this (second) scan, vectors which label only 0 or 1 population vectors are replaced by the population vectors that are most poorly represented (= largest coding error Δ)

2.2.2 Algorithm R (Replication Algorithm) :

After each complete scan of the image, the new codebook is used to determine the code label for each population vector, and the error for representation of each population vector. The most popular code vector is replaced by two equal vectors. During ensuing VQ iteration update by competitive learning (eg c-means) chose at random which of the two initially equal is the winner. This algorithm used in [10]

2.3 Image and Codebook Specifications

Here we report on experiments in the use of progressive c-means algorithm applied to the classic 256x256x8-bit gray-scale Lena image, using a (large) code-book of 256 4x4 code vectors. We consider two easily duplicated starting vector sets.

T: (Thumbnail) Set: v[q] (q = 0 .. 255) is a block of 4x4 pixels each of gray-scale q.

Q Set : Each v[q] (q = 0 .. 255) is a block of 4x4 pixels which are either 0 or 255. The first and second column, and the third and fourth column of v[q] are identical. The least significant 8-bits in the first and third column of v[q] are the binary representation of q. e.g.., $v[AA_H]$ = {(0,0,FF,FF), (0,0,FF,FF), (0,0,FF,FF), (0,0,FF,FF)}

Lena coded with (initial) Q code vectors.
PSNR = 8.7074 dB.
Very vulnerable to gerrymander:
154 vectors used in the coding;
102 vectors not used, 63 used once
Electorate statistics for electorates with population > 50:

hist[0] = 814
hist[127] = 831
hist[128] = 746
hist[255]= 814

After 10 iterations of progressive Hard C-means
PSNR = 23.4472 dB. Gerrymander established.
Gerrymander statistics:
Only 10 vectors actually utilised in this coding;
246 vectors not used, 0 used once
Electorate statistics for electorates with population > 50:

hist[0] = 692
hist[127] = 996
hist[128] = 461
hist[255] = 527
hist[7] = 743
hist[96] = 113
hist[224] = 308
hist[240] = 159

	Q Set Initial Code Vectors PSNR NU = Number Code Vectors Not used/Used Once				T Set Initial Code Vectors PSNR and NU = Number of unused Code Vectors			
I	C	CS	CR	CSR	C	CS	CR	CSR
0	8.8704	8.7074	8.7074	8.7074	23.8010	23.8010	23.8010	23.8010
NU	102/63	102/63	102/63	102/63	69/4	69/4	69/4	69/4
1	21.9802	21.9802	21.9802	21.9802	26.9345	26.9187	26.9314	26.9345
2	22.1096	23.5211	22.1096	23.9713	27.6813	28.2976	26.6730	28.2584
3	21.7651	25.9626	21.7651	25.7697	28.0389	28.5918	27.9728	28.6014
4	22.8751	26.4715	22.8751	26.0664	28.0061	28.6166	27.9140	28.7192
5	23.3482	26.6050	23.3482	26.7406	28.2682	28.6820	28.1707	28.7426
6	22.6010	26.8630	22.6010	26.9732	28.0655	28.6297	27.9965	28.6573
7	23.2983	27.0492	23.2983	27.1894	28.2297	28.6409	28.2898	28.7216
8	22.6098	27.3747	22.6098	27.3071	28.2123	28.6187	28.1418	28.6326
9	23.4542	27.3581	23.4542	27.3910	28.3252	28.6684	28.2085	28.7056
10	23.4472	27.3851	23.4472	27.5697	28.2319	28.6257	28.2165	28.6583
NU	246/0	121/27	246/0	123/26	58/61	29/71	57/58	24/69

TABLE I: Evolution of progressive c-means VQ applied to two different initial code sets versus iteration number.

3 Experimental Results

Prototypical experimental results are presented in Figs 1 and 2, and table I. For initial code vector set Q, under the progressive c-means algorithm, (C), gerrymander set in readily, and code converged to a gerrymander state with just 10 code vectors actually in use. If the additional drone eliminating algorithm S was applied, as per column CS, a better PSNR was obtained, the gross gerrymander was avoided, but not all code vectors were utilised at iteration 10. The replication algorithm by itself made the gerrymander a little worse, but was able to improve performance of the algorithm S. For initial code vector set T, for which the initial coding produces an exploded thumbnail image, the progressive c-means algorithm converges satisfactorily, with a modest decrease in the number of unused code vectors. Applying algorithm S, speeds up convergence, and significantly (0.4 dB) improves PSNR. Replication of most popular code vector, Algorithm R, has little effect on the evolution of progressive c-means, but applied in conjunction with Algorithm S has a very small enhancing effect.

4- Conclusions

The code vector histogram introduced here gave significant insight into the performance of vector quantization algorithms. We used the code histogram to identify what we termed a gerrymander - when a minority of code vectors exclude the majority of code vectors from representation. Experimental data on the VQ of gray-scale images using progressive (hard) c-means show that the replacement of the laziest code vectors (with histogram values of zero or 1), can block the development of a gerrymander; for better chosen initial vectors the same scheme speeds up convergence and improves the PSNR obtained. Duplication of the most popular code vector, as proposed by others [10], results in only a marginal improvement in VQ performance.

Although the emphasis in this paper has been on image coding, these results have special significance in areas of ANN where VQ has been used, notably in the structure and training of self-organising feature maps.[7].

It is important to recognise that the various proofs of convergence of VQ algorithms only establish convergence to local minima or saddle point of the objective function. See, e.g., [13]

Lena coded with uniform gray-scale code vectors
PSNR = 23.8010
187 vectors utilised in coding
69 not used 4 used once only
3 code vectors used > 50 times

Lena initially coded with uniform gray-scale blocks after 10 iteration of progressive c-means.
PSNR = 28.2319
198 vectors utilised in coding
58 not used 61 used once only
17 code vectors used >50 times

Lena initially coded with uniform gray-scale blocks after 10 iterations of progressive c-means with S and R. PSNR = 28.6583
215 vectors utilised in coding
41 not used 60 used once only
23 code vectors used > 50 times

Fig 2: VQ starting from uniform block code vectors.

REFERENCES

[1] S.P. Lloyd, “Least squares quantization in PCM”, Bell Labs Memo, July 1957; IEEE Trans Info Theory, IT-28, 129-137 (1982)

[2] J.Max, “Quantizing for minimum distortion”, IRE Trans. Info. Theory, IT-6(1)7-12 (1960).

[3] Y. Linde, A. Buzo, and R.M. Gray, “An algorithm for vector quantizer design”, IEEE Trans. Commun., COM-28(1), 84-95 (1980)

[4] R.M. Gray, “Vector quantization”, IEEE ASSP Magazine, 1(2), 4-29 (1984)

[5] R. Duda and P. Hart, “Pattern Classification and Scene Analysis”, Wiley, New York, 1973

[6] J. Bezdek, “Pattern Recognition with Fuzzy Objective Function Algorithms”, Plenum, NY, 1981.

[7] J.C. Dunn, "A Fuzzy Relative of the ISODATA Pocess and its Use in Detecting Compact Well-Separated Clusters", J. Cybernetics, Vol 3, No 3, pp 32-57, 1973.

[8] N.B. Karayiannis and Pin-I Pai, "A Fuzzy Algorithm for Learning Vector Quantization" Proc. IEEE Int'l Conf Systems, Man, and Cybernetics, San Antonio, Texas, Oct 2-5, 1994, Vol. 1, pp 126-131.

[9] N.B. Karayiannis, J.C. Bezdek, N.R. Pal, R.J. Hathaway, P-I Pai, “A new family of competitive learning schemes”, Submitted to IEEE Trans. on Neural Networks, July, 1995.

[10] T.Uchiyama and M.A. Arbib, "Color Image Segmentation Using Competitive Learning”, IEEE Trans. Pattern Anal. Machine Intell., vol 16, no.12, pp. 1197- 1206 , 1994.

[11] J.R. Rankin, “Application of a New Clustering Algorithm to Colour Compression”, Proc. Workshop on Electronic Colour Imaging and Applications, APRS, Canberra ACT December 1994, pp145-148

[12] T. Kohonen, “Self-organizing maps: optimization approach”, T. Kohonen, K. Makisara, O. Simula and J. Kangas (eds), Artificial Neural Networks, Elsevier Sci. Pub, 981-990 (1991).

[13] J.C. Bezdek, R. Hathaway, Michael Sabin, W. Tucker, “Convergence Theory for Fuzz c-Means: Counterexamples and Repairs”, IEEE SMC -17, No 5, pp 873-877 Sept/Oct 1987.

(b)