A FAST VQ CODEBOOK DESIGN ALGORITHM FOR A LARGE NUMBER OF DATA

M. Nakai †, *H. Shimodaira* † *and M. Kimura* ‡

† Dept. of Information Eng., Faculty of Eng., Tohoku University, Sendai-shi, 980 Japan

‡ Japan Advanced Institute of Science and Technology, East, Kanazawa-shi, 920 Japan

ABSTRUCT

This paper points out the problem that the LBG algorithm requires a lot of computation as the training vectors increase, and proposes a fast VQ (vector quantization) algorithm for a large number of training data. This algorithm consists of three steps: First, divide training vectors into some small groups; Secondly, quantize each group into a few codewords by LBG algorithm; Finally, construct a codebook by clustering these codewords using the LBG algorithm again. This paper also reports that we can reduce the distortion error of the new algorithm by adapting some effective data-dividing method. In experiments of quantizing 17,500 training vectors into 512 codewords, this algorithm requires only 1/6 computation time compared with the conventional algorithm, while the increase of distortion is only 0.5dB.

1. INTRODUCTION

The LBG vector quantizing algorithm [1] is very famous and widely used for designing VQ codebooks in many fields such as signal processing, speech processing and image processing. But it requires a large number of iterations to fix codewords with less displacement in each quantizing level. Generally, the more the number of training vectors increases, the more computation time is required. The aim of this study is to develop a fast and effective algorithm of designing a VQ codebook for a large number of training vectors.

There has been already proposed another fast algorithm for specific use in word speech recognition [2]. Where, word-specific codebooks are at first designed for each training word data set by using the LBG clustering algorithm. And next, the universal-codebook is designed by applying the LBG algorithm again to the word-specific codebooks. But the problem of this algorithm is that we do not know what kind of division of data set is adequate or optimal when considered of other general kinds of data set such as image data rather than word data.

To solve this problem, our new algorithm makes use of statistics of training data and determines more suitable division of data. Although it is somewhat similar to the algorithm above and still using the LBG algorithm twice, it can be applied to any kind of multi-dimensional data and shows better performance in quantizing distortion.

In this paper, this algorithm of using the LBG algorithm twice is named *Two-Stage Vector Quantization*, and the conventional vector quantization is named *One-Stage Vector Quantization.*

2. COMPUTATION COMPLEXITY OF THE LBG ALGORITHM

Before describing Two-Stage VQ algorithm, we had better mention the computation complexity of the conventional LBG clustering algorithm briefly. Considering the case of clustering N training vectors into 2^{i-1} codewords, it requires $2^{i-1}N$ times to calculate the distance between all training vectors and all codewords for each loop of the algorithm. So the computation complexity for clustering N training vectors into 2^{L-1} codewords is

$$N\sum_{i=1}^{L}\{2^{i-1}I(i)\} \quad .$$

Where $I(i)$ is the number of iteration required for the i-th quantizing level, but it is not predictable ahead, because the iteration for obtaining the optimal codeword continues until the decreasing rate of the quantizing distortion becomes smaller than the pre-specified threshold value. From the expression, we could make the algorithm faster by reducing either or both N, L and $I(i)$.

0-7803-0532-9/92 $3.00

3. TWO-STAGE VECTOR QUANTIZATION ALGORITHM

The basic idea of Two-Stage VQ is to reduce training vectors by dividing the original data set into small groups such as word sets. As is shown in Fig.1, this algorithm consists of the two steps of clustering called 1stVQ and 2ndVQ.

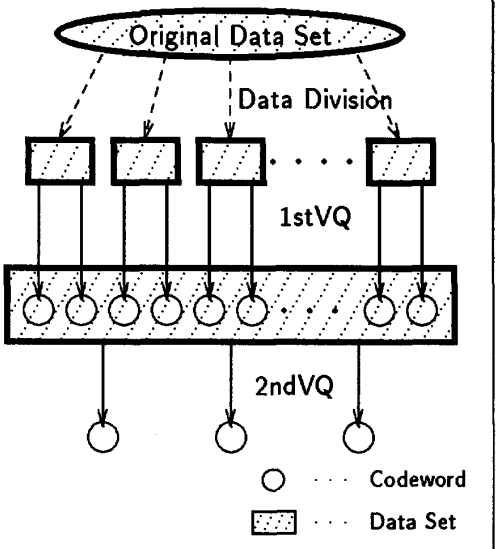

Fig.1: Conceptional diagram of Two-Stage VQ

3.1. 1stVQ

Suppose we want to design 2^{L-1} codewords from N training vectors. The N training vectors are at first divided into the G groups according to some statistics of the data set described later. Then, the training vectors of each group are quantized into 2^{P-1} codewords by the LBG clustering algorithm, where $P \leq L$. Because the number of training vectors for each group is reduced to N_j, $j = 1, 2, \cdots, G$ (N/G on the average), the total computation complexity of 1stVQ is given by

$$\sum_{j=1}^{G} \sum_{i=1}^{P} \{2^{i-1} N_j I_j(i)\} \quad ,$$

where $I_j(i)$ is the number of iteration for clustering j-th group. As we have mentioned before, $I(i)$ is not predictable ahead and it varies according to the characteristic of each data group. But from our experience it seems reasonable to suppose that the number of iteration decreases with the number of training vectors. Therefore if $I_j(i) \leq I(i)$ holds, the previous expression is bounded by

$$N \sum_{i=1}^{P} \{2^{i-1} I(i)\} \quad .$$

3.2. 2ndVQ

As a result of 1stVQ, training vectors have been reduced to $2^{P-1}G$. These are good approximations to the original training vectors with a little loss of information. Then these $2^{P-1}G$ codewords are taken as new training vectors for 2ndVQ instead of the original N training vectors. Hence, the computation complexity for 2ndVQ is

$$2^{P-1} G \sum_{i=1}^{L} \{2^{i-1} I(i)\} \quad .$$

Being reduced the number of training vectors from N to $2^{P-1}G$, it is clear that the computation time for 2ndVQ is faster than that of One-Stage VQ. Although computation complexity can be reduced very small by decreasing P and G, the following enequality should be satisfied to keep training data enough for 2ndVQ.

$$2^{P-1} G \gg 2^{L-1} \quad .$$

Finally, the computation complexity is reduced to

$$N \sum_{i=1}^{P} \{2^{i-1} I(i)\} + 2^{P-1} G \sum_{i=1}^{L} \{2^{i-1} I(i)\} \quad .$$

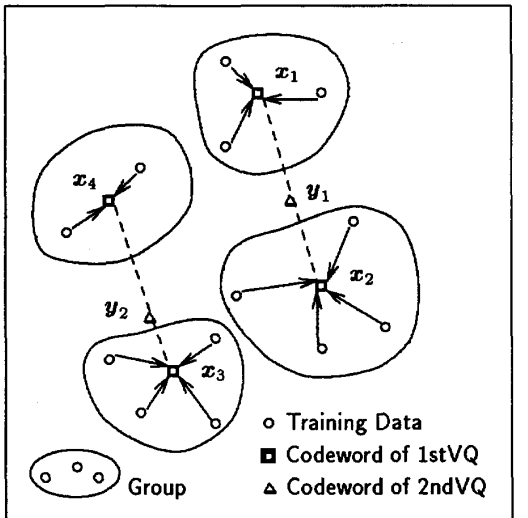

Fig.2: Weighted distortion measure of 2ndVQ

In addition, it is important to reduce the quantizing error in 2ndVQ. So we took into account how many members of the original data set are included in each cluster when we calculate each centroid vector. For example, y_1 and y_2 in Fig.2 are given by the following expressions:

$$y_1 = (3x_1 + 4x_2)/7$$
$$y_2 = (4x_3 + 2x_4)/6 \ .$$

3.3. Data Division

• Time Division

In the case of processing speech signal, the conventional fast VQ algorithm in[2] divides the training vectors into the word groups. It has been shown by experiments that only small number of codewords are enough for each group to attain reasonable distortion errors. This is because that each word consists of only a few phonemes and the training vectors from the word group are strongly dependent each other.

We call such a kind of method which divides data sequence by the order of occurrence *"TD-W"*. Fig.3 illustrates an example of dividing N vectors of M dimensions.

• Space Division

Considering another group division algorithm for more general kind of data which do not have the time-series structure like word speech, it is desirable for each group not to have the same points each other in the multi-dimensional space. Fig.4 shows three methods of dividing 2-dimensional space into 4.

The first one is the simplest method called *"SD-G,"* in which each coordinate's axis is divided at its center of gravity.

The second one is called *"SD-V"*. It is important for achieving effective clustering that the number of training vectors of each group is not biased across the groups. If there are great difference between the variances of two axes, the larger one of them is divided into 4 and the other is not, otherwise, both are divided into 2 such as the *SD-G* method.

But if the two axes correlate each other, it is no good using the *SD-V* method to reduce the bias across the groups. So the last method *"SD-P"* is proposed as more effective method for the correlated data set. This method uses the K-L transformation, which uses the principal component analysis, to decorrelate the coordinates' axes. After transforming the coordinates, each coordinate's axis is divided by the *SD-V* method.

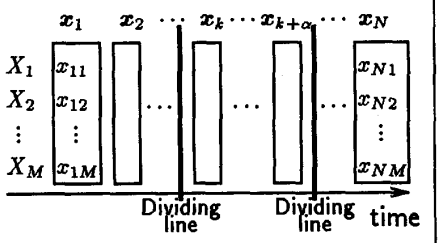

Fig.3: TD-W Method

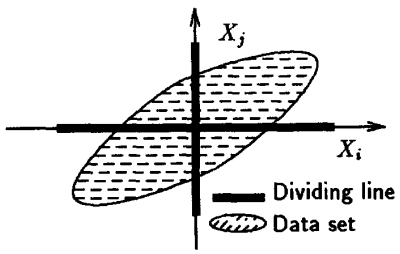

Fig.4(a): SD-G Method

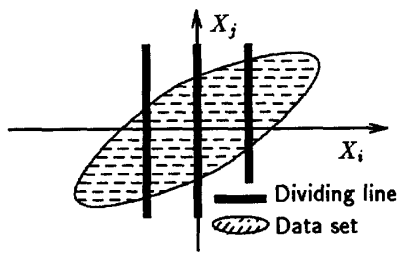

Fig.4(b): SD-V Method

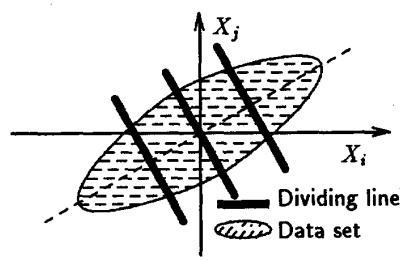

Fig.4(c): SD-P Method

4. EXPERIMENTS

Evaluation tests of new algorithm were carried out using word speech database of 212 words vocabulary under following conditions.

- **Training Vectors**
  Cepstrum 10-dimensional vectors
  (10 male speakers)
- **Quantizer Condition**
  Number of training vectors N = 17,500/speaker
  Number of divided groups $G = 1024(=2^{10})$
  Quantization level for 1stVQ $P = 3(2^2$ codewords)
  Quantization level for 2ndVQ $L = 10(2^9$ codewords)
- **Division Method**
  1) *"TD-W"* ··· Used time-series structure (conventional method).
  2) *"D-W"*··· Divide at random, not using time-series structure and multi-dimensional space.
  3) *"SD-G"* ··· Divide each axis at its centroid.
  4) *"SD-V"* ··· Vary the number of axis division according to its variance.
  5) *"SD-P"*··· Transform the coordinates by principal component analysis, and divide by *SD-V* method.
- **Result**
  It can be seen from Table 2 that if L is large enough, Two-Stage VQ requires only 1/6 computation time compared with One-Stage VQ. Table 1 shows the *SNR*(signal to noise ratio) [3] for each method. The *SNR* is considered to be a normalized quantizing distortion, and its definition is given by following expression:

$$SNR = 10\log_{10}\frac{||\boldsymbol{x} - E(\boldsymbol{x})||^2}{||\boldsymbol{x} - \hat{\boldsymbol{x}}||^2} \quad ,$$

where $\hat{\boldsymbol{x}}$ is the codeword of each data $\boldsymbol{x}$, and $||\boldsymbol{x} - \hat{\boldsymbol{x}}||^2$ (the least squared error) represents a total distortion. It can be found from the table that the *SD-P* method achieved the highest *SNR* among the five dividing methods.

5. CONCLUSION

We have developed a fast algorithm named "Two-Stage VQ" for designing a VQ codebook from a large number of data. The basic idea of realizing fast VQ is to divide training vectors into small groups by using the characteristics of vectors either in time-domain or in space-domain before the conventional VQ processing. In experiments, this algorithm reduced the computation complexity by a factor of 6. And we showed that data-dividing methods in space-domain, especially *SD-P* method which uses K-L transformation, are useful than those in time-domain for achieving low distortion VQ. Although experiments were done by using speech data, the proposed algorithm is applicable to other fields such as image coding.

References

[1] Y. Linde, A. Buzo and R. M. Gray, "An Algorithm for Vector Quantizer Design," *IEEE Trans.* Vol.COM-28 (1980-01)

[2] S. Furui, "A VQ-Based Preprocessor Using Cepstral Dynamic Feature for Large Vocabulary Word Recognition," *Proc.ICASSP 87*, pp.1127-1130 (1987)

[3] R. M. Gray, "Vector Quantization," *IEEE ASSP Magazine* (1984-04)

Table 1: SNR of each VQ method

level	1-Stage	2-Stage VQ method				
		TD-W	D-R	SD-R	SD-V	SD-P
1(1)	*0.000*	0.000	0.000	0.000	0.000	0.000
2(2)	*1.902*	**1.902**	1.891	1.899	1.899	1.900
3(4)	*3.536*	**3.519**	3.309	3.508	3.514	3.489
4(8)	*5.351*	**5.298**	4.853	5.226	5.282	5.338
5(16)	*6.476*	**6.449**	5.792	6.418	6.439	6.428
6(32)	*7.431*	7.355	6.542	7.323	7.349	**7.361**
7(64)	*8.321*	8.182	7.201	8.148	8.159	**8.198**
8(128)	*9.191*	8.950	7.800	8.917	8.945	**8.980**
9(256)	*10.069*	9.694	-	-	9.699	**9.748**
10(512)	*11.011*	10.444	-	-	10.463	**10.532**

(dB)

Table 2: Execution Time

level	1-StageVQ	2-Stage VQ	rate
1(1)	1.00	9.07	9.07
2(2)	4.62	10.00	2.16
3(4)	12.43	12.34	0.99
4(8)	35.48	18.79	0.53
5(16)	95.90	29.96	0.31
6(32)	224.94	55.77	0.24
7(64)	471.16	104.30	0.22
8(128)	954.22	188.42	0.20
9(256)	1782.33	321.91	0.18
10(512)	3197.19	550.24	0.17

(normalized by 1-Stage, 1-level)