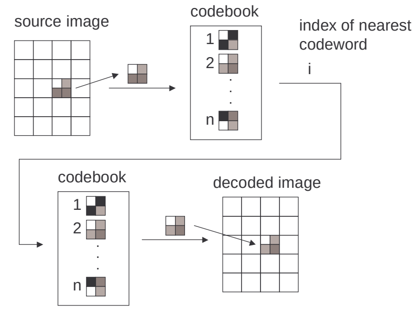
CSE 490 GZ
Introduction to Data Compression
Winter 2004

Vector Quantization

Vector Quantization



Vectors

- An a x b block can be considered to be a vector of dimension ab.

block
w	x
y	z

= (w,x,y,z) vector

- Nearest means in terms of Euclidian distance or Euclidian squared distance. Both equivalent.

$$\text{Distance} = \sqrt{(w_1 - w_2)^2 + (x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}$$

$$\text{Squared Distance} = (w_1 - w_2)^2 + (x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2$$

- Squared distance is easier to calculate.

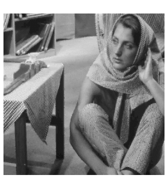
Vector Quantization Facts

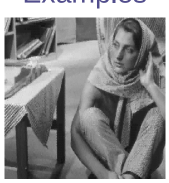
- The image is partitioned into a x b blocks.
- The codebook has n representative a x b blocks called codewords, each with an index.
- Compression with fixed length codes is

$$\frac{\log_2 n}{ab} \text{ bpp}$$

- Example: a = b = 4 and n = 1,024
  - compression is 10/16 = .63 bpp
  - compression ratio is 8 : .63 = 12.8 : 1
- Better compression with entropy coding of indices

Examples


4 x 4 blocks
.63 bpp


4 x 8 blocks
.31 bpp


8 x 8 blocks
.16 bpp

Codebook size = 1,024

Scalar vs. Vector

- Pixels within a block are correlated.
  - This tends to minimize the number of codewords needed to represent the vectors well.
- More flexibility.
  - Different size blocks
  - Different size codebooks

Encoding and Decoding

- Encoding:
  - Scan the a x b blocks of the image. For each block find the nearest codeword in the codebook and output its index.
  - Nearest neighbor search.
- Decoding:
  - For each index output the codeword with that index into the destination image.
  - Table lookup.

The Codebook

- Both encoder and decoder must have the same codebook.
- The codebook must be useful for many images and be stored someplace.
- The codebook must be designed properly to be effective.
- Design requires a representative training set.
- These are major drawbacks to VQ.

Codebook Design Problem

- Input: A training set X of vectors of dimension d and a number n. (d = a x b and n is number of codewords)
- Ouput: n codewords c(0), c(1),...,c(n-1) that minimizes the distortion.

$$D = \sum_{x \in X} \|x - c(\text{index}(x))\|^2 \quad \text{sum of squared distances}$$

where index(x) is the index of the nearest codeword to x.

$$\|(x_0, x_1 \cdots, x_{d-1})\|^2 = x_0^2 + x_1^2 + \cdots + x_{d-1}^2 \quad \text{squared norm}$$

GLA

- The Generalized Lloyd Algorithm (GLA) extends the Lloyd algorithm for scalars.
  - Also called LBG after inventors Linde,Buzo, Gray (1980)
- It can be very slow for large training sets.

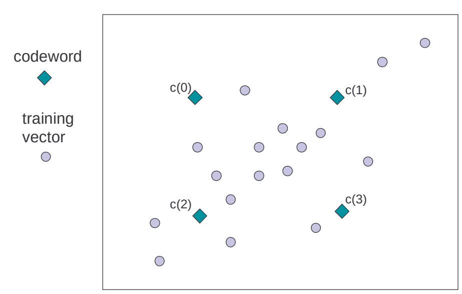
GLA

```
Choose a training set X and small error tolerance ε > 0.
Choose start codewords c(0),c(1),...,c(n-1)
Compute X(j) := {x : x is a vector in X closest to c(j)}
Compute distortion D for c(0),c(1),...,c(n-1)
Repeat
   Compute new codewords
```

$$c'(j) := \text{round}\left(\frac{1}{|X(j)|} \sum_{x \in X(j)} x\right) \quad \text{(centroid)}$$

```
   Compute X'(j) = {x : x is a vector in X closest to c'(j)}
   Compute distortion D' for c'(0),c'(1),...,c'(n-1)
    if |(D – D')/D| < ε then quit
    else c := c'; X := X', D := D'
End{repeat}
```

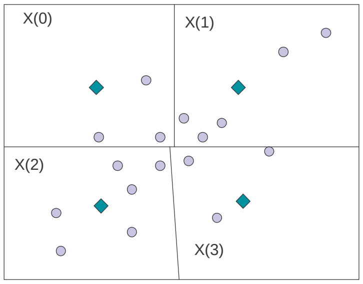

GLA Example (1)



GLA Example (2)

codeword

training vector

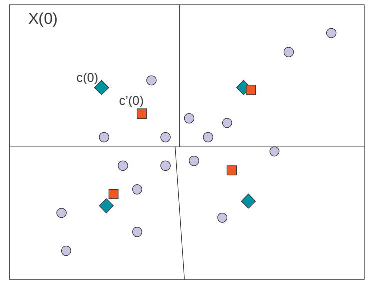


GLA Example (3)

codeword

training vector

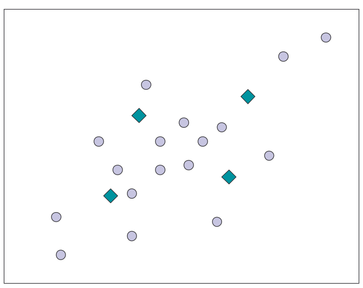
centroid



GLA Example (4)

codeword

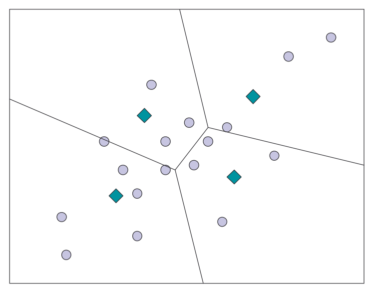
training vector



GLA Example (5)

codeword

training vector

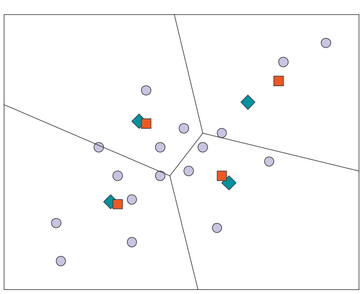


GLA Example (6)

codeword

training vector

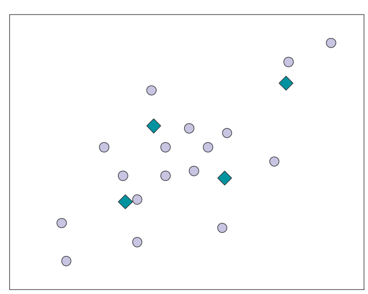
centroid



GLA Example (7)

codeword

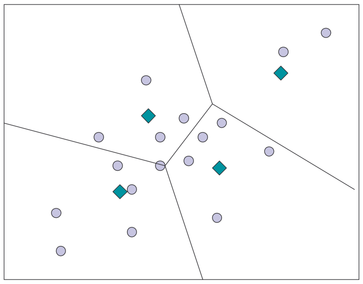
training vector



GLA Example (8)

codeword

training vector

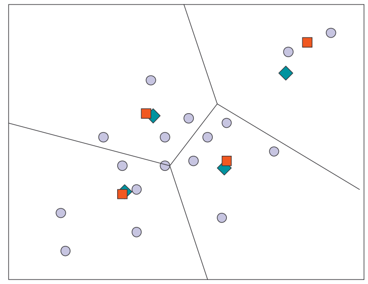


GLA Example (9)

codeword

training vector

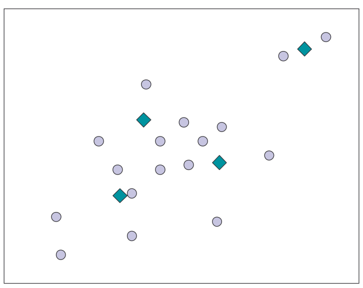
centroid



GLA Example (10)

codeword

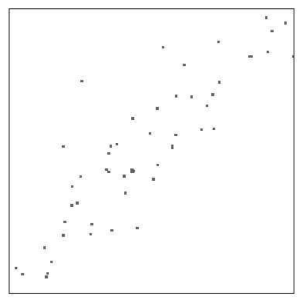
training vector



Codebook

1 x 2 codewords

Note: codewords diagonally spread



Codeword Splitting

- It is possible that a chosen codeword represents no training vectors, that is, X(j) is empty.
  - Splitting is an alternative codebook design algorithm that avoids this problem.
- Basic Idea
  - Select codeword c(j) with the greatest distortion.

$$D(j) = \sum_{x \in X(j)} \|x - c(j)\|^2$$

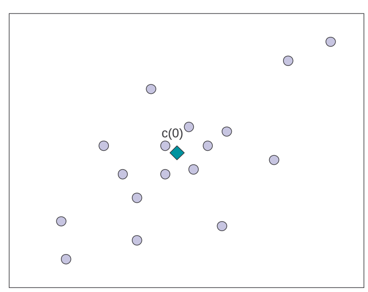
  - Split it into two codewords then do the GLA.

Example of Splitting

codeword

training vector

Initially c(0) is centroid of training set

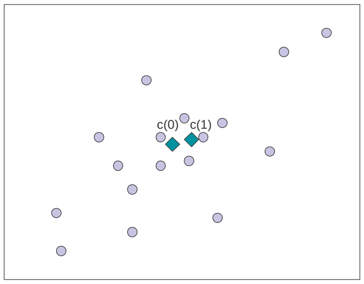


Example of Splitting

codeword

training vector

Split
$c(1) = c(0) + \varepsilon$

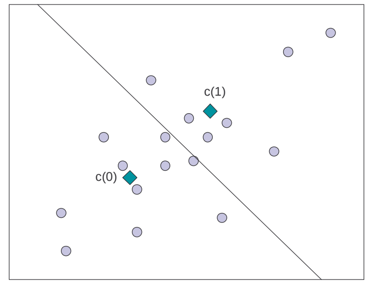


Example of Splitting

codeword

training vector

Apply GLA

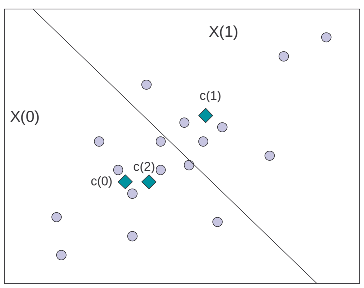


Example of Splitting

codeword

training vector

c(0) has max distortion so split it.

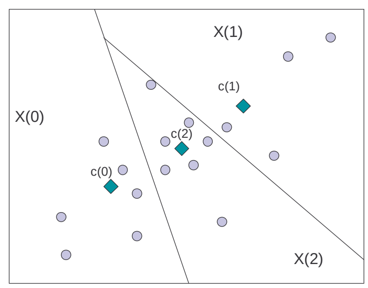


Example of Splitting

codeword

training vector

Apply GLA

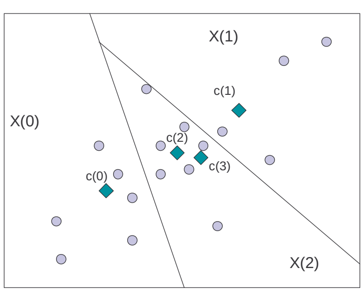


Example of Splitting

codeword

training vector

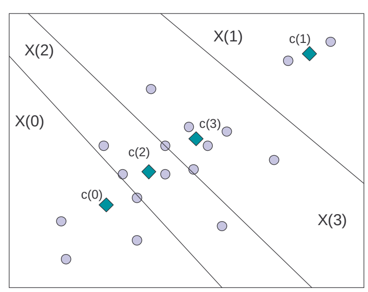
c(2) has max distortion so split it



Example of Splitting

codeword

training vector



GLA Advice

- Time per iteration is dominated by the partitioning step, which is m nearest neighbor searches where m is the training set size.
  - Average time per iteration O(m log n) assuming d is small.
- Training set size.
  - Training set should be at least 20 training vectors per code word to get reasonable performance.
  - To small a training set results in "over training".
- Number of iterations can be large.

Encoding

- Naive method.
  - For each input block, search the entire codebook to find the closest codeword.
  - Time O(T n) where n is the size of the codebook and T is the number of blocks in the image.
  - Example: n = 1024, T = 256 x 256 = 65,536 (2 x 2 blocks for a 512 x 512 image)
    nT = 1024 x 65536 = $2^{26} \approx$ 67 million distance calculations.
- Faster methods are known for doing "Full Search VQ". For example, k-d trees.
  - Time O(T log n)