CSE 589
Applied Algorithms
Spring 1999

Image Compression
Vector Quantization
Nearest Neighbor Search

Lossy Image Compression Methods

- Basic theory - trade-off between bit rate and distortion.
- Vector quantization (VQ).
  - A indices of set of representative blocks can be used to code an image, yielding good compression. Requires training.
- Wavelet Compression.
  - An image can be decomposed into a low resolution version and the detail needed to recover the original. Sending most significant bits of the wavelet coded yields excellent compression.

JPEG Standard

- JPEG - Joint Photographic Experts Group
  - Current image compression standard. Uses discrete cosine transform, scalar quantization, and Huffman coding.
  - JPEG 2000 will move to wavelet compression.

Barbara


original


JPEG

32:1 compression ratio
.25 bits/pixel (8 bits)


VQ


Wavelet-SPIHT

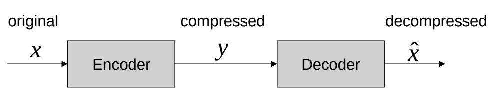
JPEG

VQ

SPIHT

Original

Distortion



- Lossy compression: $x \neq \hat{x}$
- Measure of distortion is commonly mean squared error (MSE). Assume x has n real components (pixels).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2$$

PSNR

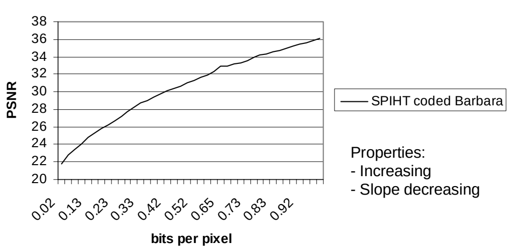
- Peak Signal to Noise Ratio (PSNR) is the standard way to measure fidelity.

$$PSNR = 10\log_{10}\left(\frac{m^2}{MSE}\right)$$

where m is the maximum value of a pixel possible.
For gray scale images (8 bits per pixel) m = 255.

- PSNR is measured in decibels (dB).
- .5 to 1 dB is said to be a perceptible difference.

Rate-Fidelity Curve

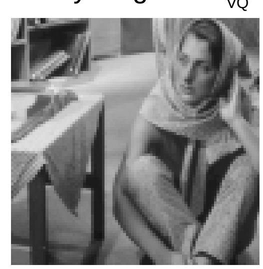


Properties:
- Increasing
- Slope decreasing

PSNR is not Everything

VQ

PSNR = 25.8 dB



PSNR = 25.8 dB

PSNR Reflects Fidelity (1)

VQ

PSNR 25.8
.63 bpp
12.8 : 1

PSNR Reflects Fidelity (2)

VQ

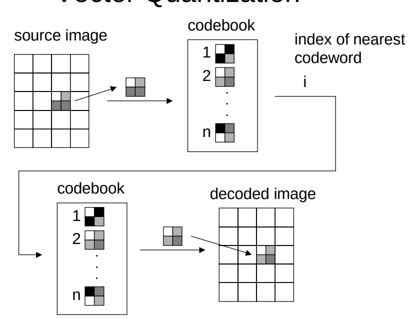
PSNR 24.2
.31 bpp
25.6 : 1

PSNR Reflects Fidelity (2)

VQ

PSNR 23.2
.16 bpp
51.2 : 1

Vector Quantization



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 is

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

- Example: $a = b = 4$ and $n = 1{,}024$
  - compression is 10/16 = .63 bpp
  - compression ratio is 8 : .63 = 12.8 : 1

Examples

4 x 4 blocks
.63 bpp

4 x 8 blocks
.31 bpp

8 x 8 blocks
.16 bpp

Codebook size = 1,024

Encoding and Decoding

- Encoding:
  - Scan the a x b blocks of the image. For each block find the nearest codeword in the code book 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 vectors $c_1, c_2, \ldots c_n$ that minimizes the sum of the distances from each member of the training set to its nearest codeword. That is minimizes

$$\sum_{x\in X}\left\|c_{n(x)}-x\right\| = \sum_{x\in X}\sqrt{\sum_{i=1}^{d}\left(c_{n(x)}(i)-x(i)\right)^2}$$

where $c_{n(x)}$ is the nearest codeword to x.

Algorithms for Codebook Design

- The optimal codebook design problem appears to be a NP-hard problem.
- There is a very effective method, called the generalized Lloyd algorithm (GLA) for finding a good local minimum.
- GLA is also known in the statistics community as the k-means algorithm.
- GLA is slow.

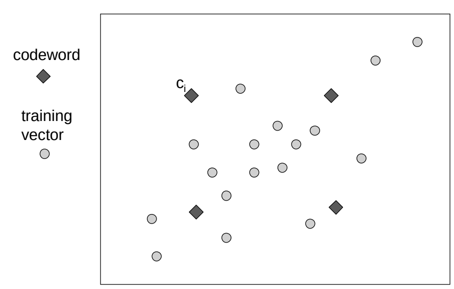
GLA

- Start with an initial codebook $c_1, c_2, \ldots c_n$ and training set X.
- Iterate:
  - Partition X into $X_1, X_2, \ldots, X_n$ where X_i includes the members of X that are closest to c_i.
  - Let x_i be the centroid of X_i .

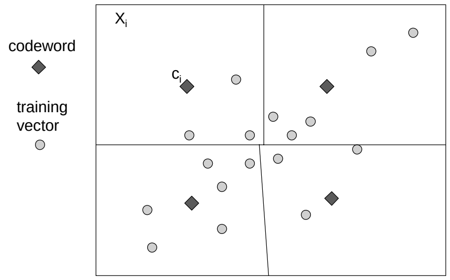
$$x_i = \frac{1}{|X_i|}\sum_{x\in X_i} x$$

  - if $\sum_{i=1}^{n}\left\|x_i - c_i\right\|$ is sufficiently small then quit. otherwise continue the interation with the new codebook $c_1, c_2, \ldots c_n = x_1, x_2, \ldots x_n$.

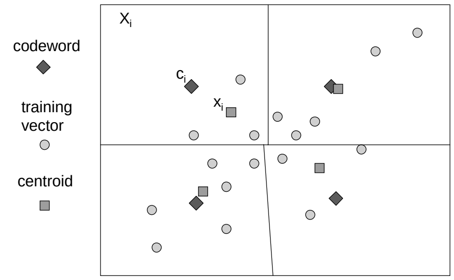
GLA Example (1)



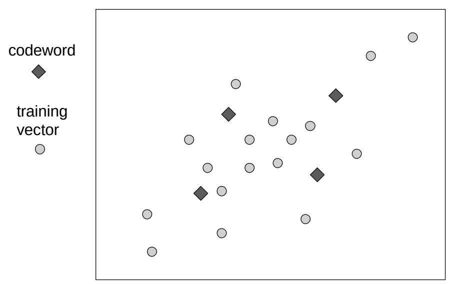
GLA Example (2)



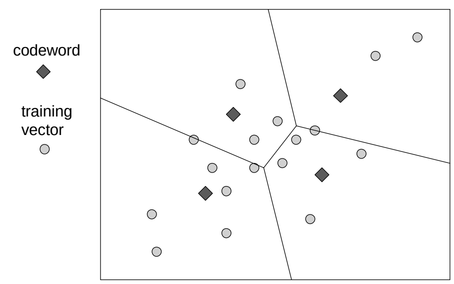
GLA Example (3)



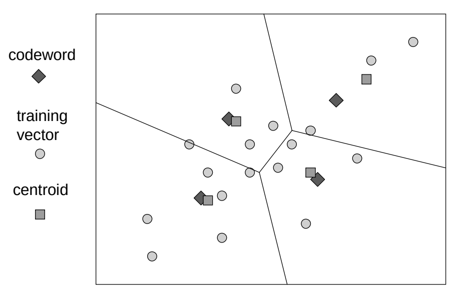
GLA Example (4)



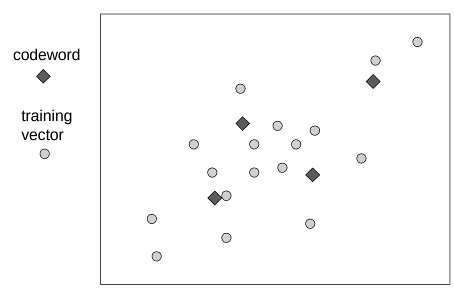
GLA Example (5)



GLA Example (6)



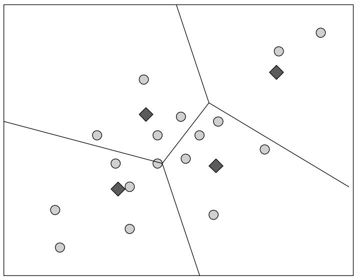
GLA Example (7)



GLA Example (8)

codeword

training vector

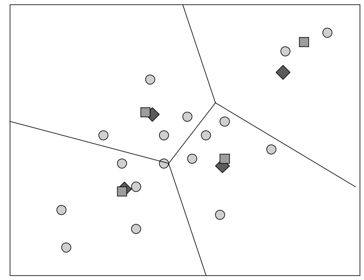


GLA Example (9)

codeword

training vector

centroid

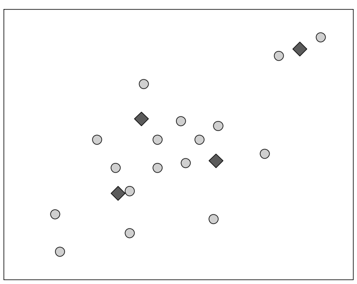


GLA Example (10)

codeword

training vector

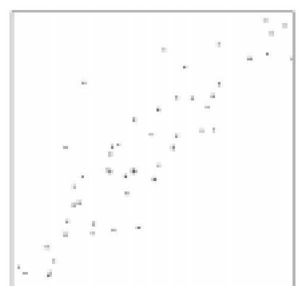
centroid



Codebook

1 x 2 codewords

Note: codewords diagonally spread



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.

Nearest Neighbor Search

- Preprocess a set of n vectors V in d dimensions into a search data structure.
- Input: A query vector q.
- Ouput: The vector v in V that is nearest to q. That is, the vector v in V that minimizes

$$\|v-q\|^2 = \sum_{i=1}^{d} \left((v(i)-q(i))\right)^2$$

NNS in VQ

- Used in codebook design.
  - Used in GLA to partition the training set.
  - Since codebook design is seldom done then speed of NNS is not too big a issue.
- Used in VQ encoding.
  - Codebook size is commonly 1,000 or more.
  - Naive linear search would make encoding too slow.
  - Can we do better than linear search?

Naive Linear Search

- Keep track of the current best vector, best-v, and best distance squared, best-squared-d.
  - For an unsearched vector v compute $||v - q||^2$ to see if it smaller than best-squared-d.
  - If so then replace best-v.
  - If d is moderately large it is a good idea not to compute the squared distance completely. Bail out when k < d and

$$\text{best} - \text{squared} - \text{d} \le \sum_{i=1}^{k} (v(i) - q(i))^2$$

Orchard's Method

- Invented by Orchard (1991)
- Uses a “guess codeword”. The guess codeword is the codeword of an adjacent coded vector.
- Orchard's Principle.
  - if r is the distance between the guess codeword and the query vector q then the nearest codeword to q must be within a distance of 2r of the guess codeword.

Orchard's Principle