Predictive Encoding

Computer Science Department
CS4481b/9628b: Image Compression
Winter 2017
Instructor: Mahmoud R. El-Sakka
Office: MC-419
Email: elsakka@csd.uwo.ca
Phone: 519-661-2111 x86996

Introduction

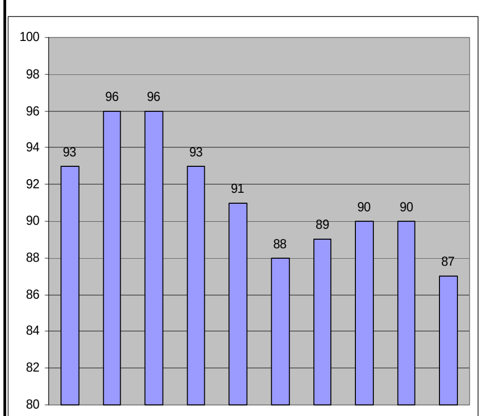
- In typical images, the values of adjacent pixels are highly correlated, i.e., a great deal of information about a pixel value can be obtained by inspecting its neighbouring pixel values
- This property is exploited in predictive encoding schemes
- This class of compression schemes is based on reducing the *inter-pixel* redundancy of closely spaced pixels by extracting and encoding ***only*** the new information in each pixel
- The ***new information*** of a pixel is defined as the difference between the *actual* and *predicted* value of the pixel, where the prediction is made based on the values of the surrounding pixels

DPCM Encoding

- By far, the *differential pulse code modulation* (DPCM) is the most common predictive encoding approach
- DPCM encoding scheme can be
  - Lossless
  - Lossy

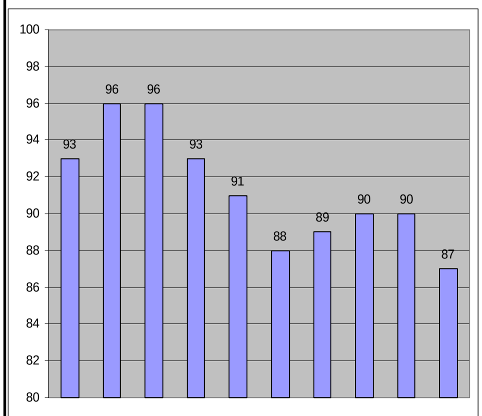
Prediction Example

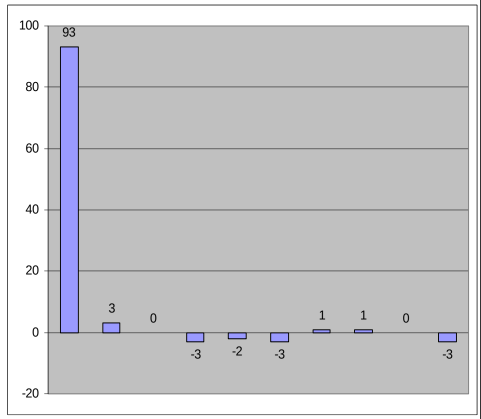
- Consider the following sequence of numbers:
  93 96 96 93 91 88 89 90 90 87



Prediction Example

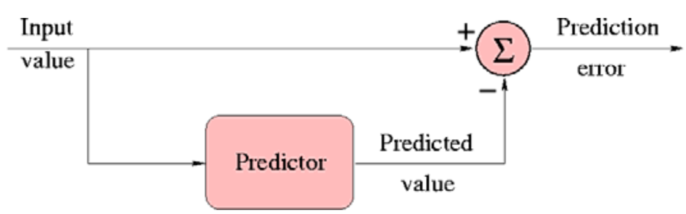
- Taking the difference between samples, assuming that the first sample value is zero, will produce the following sequence: 93 3 0 -3 -2 -3 1 1 0 -3





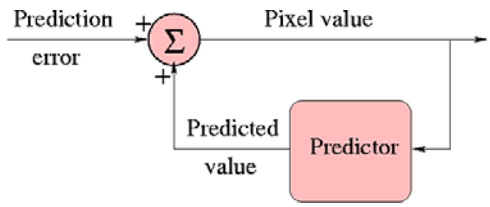
Prediction Encoding

- Input: 93 96 96 93 91 88 89 90 90 87
- Output: 93 3 0 -3 -2 -3 1 1 0 -3



Prediction Decoding

- Input: 93 3 0 -3 -2 -3 1 1 0 -3
- Output: 93 96 96 93 91 88 89 90 90 87



DPCM Encoding/Decoding Scheme

- Each of the *encoder* and the *decoder* contains an identical *predictor*
  - As each successive *pixel of the input image* is introduced to the *encoder*, the predictor generates the *anticipated value* of that pixel, based on some past pixel values
  - This predicted value is used to generate *the prediction error*
  - The *decoder* uses this prediction error value to *reconstruct the original pixel value*

Prediction

- Current pixel *can be predicted* as a function of *already-encoded* surrounding pixels
  why already-encoded pixels only?
- The prediction function can simply be a *linear combination* (*weighted sum*) of previously encoded-pixel values, i.e.,

$$\hat{f}(x,y)=\sum_{k=0}^{y} w_{x,k}\times\hat{f}(x,y-k)+\sum_{i=1}^{x}\sum_{j=0}^{image\_width-1} w_{x-i,j}\times\hat{f}(x-i,j)$$

where:
  - $\hat{f}(x,y)$ is the predicted value
  - $w_{x,k}$ and $w_{x\text{-}i,j}$ are weighting factors
  - $\hat{f}(x,y-k)$ and $\hat{f}(x-i,y)$ are previously encoded pixel values

Topic 08: Predictive Encoding

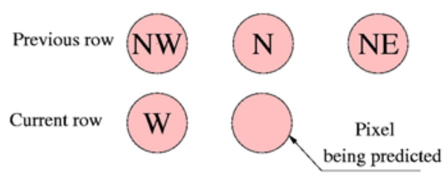
Prediction

- Since the values of adjacent pixels are highly correlated, the prediction should mainly rely on *already-encoded* surrounding pixels

Prediction

- The ***number of pixels used*** in a predictor is called the ***order*** of the predictor
- The prediction is referred to as
  - ***One-dimensional prediction***, if only pixels from the current scan row, ***or*** column, are utilized in forming the prediction
  - ***Two-dimensional prediction***, if pixels from the previous scan row(s) ***and*** column(s) are utilized in forming the prediction

Prediction



- Examples:
  - First-order one-dimensional predictor

$$\hat{f}(x,y) = W \qquad \hat{f}(x,y) = N$$

  - Second-order two-dimensional predictor

$$\hat{f}(x,y) = 0.75 \times W + 0.25 \times N$$

$$\hat{f}(x,y) = 1.25 \times W - 0.25 \times N$$

  - Third-order two-dimensional predictor

$$\hat{f}(x,y) = 0.75 \times W + 0.75 \times N - 0.5 \times NW$$

Prediction

- A higher order predictor would outperform ones of lower order, *but* it is more complicated

- A two-dimensional prediction leads to improved results, as compared to one-dimensional prediction,
but a two-dimensional prediction requires buffering pixel values in previous rows

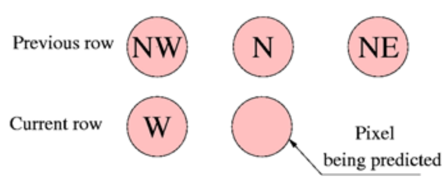
Prediction

- The set of predictor coefficients (weighting factors) may be
  - Fixed for all images (*global* prediction)
  - Vary from image to image (*local* prediction)
  - Vary within an image to accommodate the local changes in image statistics (*adaptive* prediction)

Prediction

- In case of *local* or *adaptive* predictions, weighting factors can be optimized by minimizing an error criterion, e.g., mean-squared prediction error
  - Optimizing the weighting factors for each image could be impractical for many real time applications
  - The selection of a robust set of *global* weighting factors for a typical class of images can be achieved by optimizing these weighting factors on a set of training images from this class

Old JPEG Standard (lossless mode)



- The lossless part of the ***old*** JPEG standard is nothing put *DPCM* followed by *Huffman encoding*
- There are seven different predictors that can be used

$$\hat{f}(x,y) = W \qquad \hat{f}(x,y) = W - 0.5\times NW + 0.5\times N$$

$$\hat{f}(x,y) = N \qquad \hat{f}(x,y) = 0.5\times W - 0.5\times NW + N$$

$$\hat{f}(x,y) = NW \qquad \hat{f}(x,y) = 0.5\times W + 0.5\times N$$

$$\hat{f}(x,y) = W - NW + N$$

Old JPEG Standard (lossless mode)

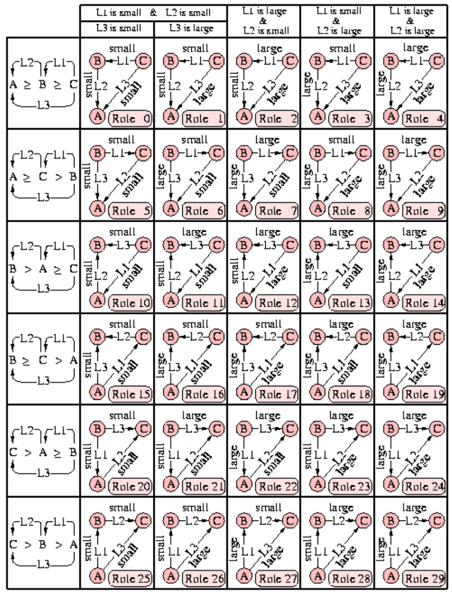
- Once a predictor is selected (by the user), the whole pixel values are predicted using this predictor
- The ***old*** JPEG does not provide adaptive prediction

Adaptive DPCM

- In adaptive DPCM, the prediction rule is chosen based on the surrounding pixel values
- In this case, we can use one, out of *n* predictors, without sending any side information to the decoder; yet the decoder can correctly identify which prediction rule was chosen

Adaptive ABC-SC

- *Example 1:*



CALIC

- *Example 2:*
  Context Adaptive Lossless Image Compression (CALIC)
- CALIC is one of the best lossless image compression scheme
  - ranked top among schemes that were evaluated by the JPEG committee in July 1995 prior to the development of the JPEG-LS standard
- CALIC is a ***sequential*** encoding scheme that encodes and decodes in a raster scan order with a single pass through the image
- CALIC codec is ***symmetric***, meaning that the encoder and the decoder have the same time and space complexity

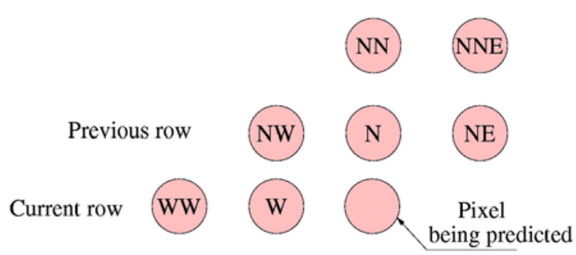
CALIC

- *Example 2:*
- CALIC operates in two modes:
  - Binary mode
  - Continuous-tone mode

- CALIC ***automatically*** selects one of the two modes on the fly during the coding process, depending on the context of the current pixel
- The *initial* prediction is based on the 7 surrounding pixel values

The basic Idea of CALIC *binary* Mode

- The algorithm checks WW, W, NW, N, NE, NN, NNE
- If these pixels have ***no more than 2 distinct values***, binary mode is triggered automatically (transparent to the user)



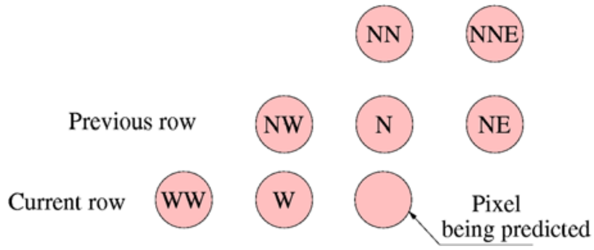
The basic Idea of CALIC *binary* Mode

- Let s_1 be the value of W, and s_2 be the other value, ***if any***
- Ternary code T is used to encode the *current pixel*
  - T = 0 if *current pixel* = s_1
  - T = 1 if *current pixel* = s_2
  - T = 2 otherwise
- T = 2 is the ***escape signal to exit binary mode*** and go back to the *continuous-tone mode*

The basic Idea of CALIC *continuous-tone* Mode

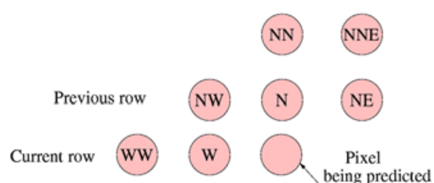
- In continuous-tone mode, there are three major components:
  - Gradient-adjusted prediction (GAP)
  - Context modeling
  - Entropy coding of prediction errors

The basic Idea of CALIC *continuous-tone* Mode



- Two gradients (d_h and d_v) are calculated:
  - d_h=|W–WW| + |N–NW| + |NE–N|
  - d_v=|W–NW| + |N–NN| + |NE–NNE|
- Using d_h and d_v, the Gradient-Adjusted Prediction (GAP) value is computed

The basic Idea of CALIC *continuous-tone* Mode




```
if (dv – dh > 80) {sharp horizontal edge}
    Prediction = W
else
    if (dh – dv > 80) {sharp vertical edge}
      Prediction = N
    else
    { Prediction = (W + N)/2 + (NE – NW)/4
      if(dv – dh > 32) {horizontal edge}
        Prediction= (1/2 × Prediction + 1/2 × W)
      else
        if (dh – dv > 32) {vertical edge}
          Prediction= (1/2 × Prediction + 1/2 × N)
        else
      if(dv – dh > 8) {weak horizontal edge}
        Prediction= (3/4 × Prediction + 1/4 × W)
      else
        if (dh – dv > 8) {weak vertical edge}
          Prediction= (3/4 × Prediction + 1/4 × N)
    }
```

The basic Idea of CALIC *continuous-tone* Mode

- The thresholds (80, 32, and 8) given in the GAP procedure are for eight bit data
- These thresholds
  - were empirically selected by the authors after extensive experimentation with a large set of test images
  - can be also
    - adapted on the fly for higher resolution images
    - Specified by the user, in case if off-line optimization is performed by the user

The basic Idea of CALIC *continuous-tone* Mode

- ***Is GAP a linear or non-leaner predictor?***
- Gap is a simple, adaptive, non-linear predictor that can adapt itself based on the gradient near the predicted pixel
- GAP is more robust than linear predictors, particularly in areas of strong edges
- GAP differs from the existing linear predictors in that it weight the neighboring pixels according to the estimated gradients

NEW JPEG Standard (lossless mode)

- The ***new*** JPEG-LS standard looks more like CALIC than the ***old*** JPEG standard
- The ***new*** JPEG-LS standard is called LOCO (stand for LOw COmplixity)
- LOCO is developed by *Hewlett-Packard (HP)*
- LOCO is a predictive scheme
- LOCO is a much simpler than CALIC, yet still performs close to CALIC