DIGITAL TELECOMMUNICATIONS

University of Patras

Department of Computer Engineering and Informatics

Academic Year 2023-2024

Part A

Information Theory

The purpose of this exercise is to understand basic concepts from the Information Theory. This will be achieved through the study of relevant literature and the implementation of a discrete source encoding/decoding system. In particular, a system will be implemented which uses Huffman coding for image compression. For the implementation as well as the experimental measurements it is proposed to use the MATLAB computing environment.

Source: The source to be used in this implementation is an image("parrot.png") which is represented as a matrix of integers $I \in Z\ N \times M$.

Questions

You are called to prepare a technical report which answers the following questions and briefly explain the methodology you have used to answer.

1. **a.** First, you are asked to identify the symbols of the source (i.e. the distinct values that the pixels of the image receive) and estimate their probabilities of occurrence.
   b. Next, compute the Huffman coding for the given source. Calculate: i) the entropy of your coding, ii) the average code length, and iii) the efficiency of your code. Comment briefly on your results.

2. Now consider the second-order source extension of question 1.

   a. You are asked to identify the symbols of the second-order source extension (pairs of characters) and for each pair estimate the probability of its occurrence.

   b. Next, compute the Huffman encoding for the given source. Calculate: i) the entropy of your encoding, ii) the average code length, and iii) the efficiency of your code. Comment briefly on your results.

   c. How do your results compare, with those of question 1?

3.

a. Explain briefly why the formula does not apply in this case:

$H(X\text{^}2) = 2H(X)$ for calculating the second order entropy source expansion.

b. List a barrier of the form $a \leq \bar{L} < b$ for the average code length that you computed in questions 1,2.

4. Encode the source using the Huffman code calculated in question 1. Verify the correctness of your encoding (e.g., through the inverse process – decoding). Compare the number of bits in the binary representation of the image with the Huffman coding by calculating the compression ratio:

J = (# of bits in Huffman coding) / (# of bits in binary representation)

5. Now, we wish to transmit the encoded sequence through a Binary Symmetric Channel for which we do not know the transition probability p. Specifically, consider the function "y = binary_symmetric_channel(x)," which models the channel and takes the codeword vector (x) as input and produces the corresponding bit sequence observed by the receiver (y). Using the two bit sequences (x, y), estimate the parameter p (to two decimal places) and compute the channel capacity. Also, compute the mutual information between the channel's input and output.

Notes:

- It is noted that for the implementation of this particular question, it is recommended to use the following MATLAB functions: huffmandict, huffmanenco, huffmandeco.
- You can compute the binary representation of the image using the following command:
  bI = reshape((dec2bin(typecast(image(:), 'uint8'), 4) - '0').', 1, [])
- To load the image file into memory, use the command:
  I = imread('parrot.png');

Part b

Coding a discrete source with the DPCM method

1. Introduction

The DPCM (Differential Pulse Code Modulation) can be considered a generalization of Delta coding, where the quantized signal sent to the receiver is the difference between the current sample (at time n) and a linear prediction of it. In DPCM, we compute, at each time point, a prediction for the value of the current sample based on the values of previous samples that have already been encoded. Then, we calculate the error of this prediction. The prediction error signal is quantized and then encoded using one or more binary digits per sample.

2. DPCM Encoding

The encoder and decoder of a DPCM system are shown in Figure 1. To quantize and encode the value of the current sample, we first calculate a prediction for its value based on the encoded values of previous samples. The predicted signal $x(n)$ is denoted as $y\hat{}'(n)$. In the figure, we observe a memory arrangement (both at the transmitter and the receiver) that stores the reconstructed values of the previous samples, based on which the prediction for the value of the current sample is calculated. Our goal is to minimize the variance of the error signal $y(n) = x(n) - y\hat{}'(n)$, so that it has a small dynamic range and can be adequately described by a small number of binary digits. The process of quantizing the error signal $y(n)$ results in the signal $y\hat{}(n)$, which is then sent to the receiver.

At the receiver, the signal $y\hat{}(n)$ is combined with the signal $y\hat{}'(n)$ (the prediction of $x(n)$). Since the previously reconstructed values and the prediction method used by the transmitter are known to the receiver, both the transmitter and the receiver can compute exactly the same prediction values $y\hat{}'(n)$. As in the case of Delta coding, the transmitter includes the receiver's arrangement, which computes the reconstruction $x\hat{}(n)$. The transmitter uses these values for prediction, rather than the actual values $x(n)$, in order to fully emulate the receiver's arrangement, which naturally does not know the actual values. By using the reconstructed values to compute the prediction and subsequently the prediction error, we ensure (as in the case of Delta coding) that there is no accumulation of quantization error.

In the simple case where we rely only on the prediction of the previous sample, the equations that describe the operation of the DPCM system in Figure 2 are as follows:
$y(n) = x(n) - y\hat{}'(n-1)$
$y\hat{}(n) = Q(y(n))$
$y\hat{}'(n) = y\hat{}(n) + y\hat{}'(n-1)$

where $Q(\cdot)$ is the input-output function of the scalar (uniform) quantizer used. From the above relationships, we derive the expression for the quantization error:
$yQ(n) = x\hat{}(n) - x(n) = y\hat{}(n) - y(n)$.

We observe that if in the above equations we set $y\hat{}'(n) = 0$, i.e., a DPCM system that does not use prediction, then this system is equivalent to a simple PCM coding system.

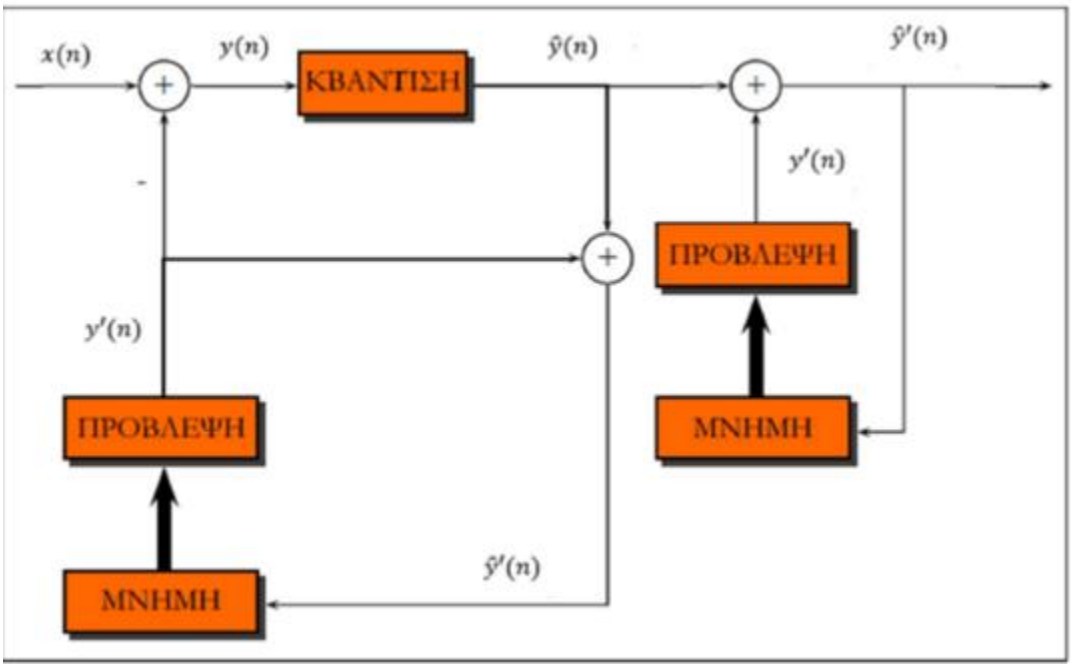

Figure 1.

3. Calculating the prediction filter

In a general DPCM system, the prediction of the sample $x(n)$ is given as a linear combination of p previous values $y\hat{}'(n-i)$ that have already been encoded and reconstructed, i.e.:

$y\hat{}'(n) = \sum pi{=}1\ aiy\hat{}'(n-i)$

Our goal is to calculate the coefficients ai by minimizing the mean square error between each current input sample and its prediction:

$MSE = E[e^2(n)] = E\ [x(n) - \sum pi{=}1\ aiy\hat{}'(n-i)]$

However, this criterion is difficult to minimize because the MSE depends on both the coefficients and the quantizer we use. Therefore, it constitutes a nonlinear minimization problem. To overcome this difficulty, we replace $y\hat{}'(n-i)$ with $x(n-i)$, assuming that the latter, being the quantized version of the former, introduces minimal error. Thus, we can minimize the following expression:

$MSE \hat{=} E[e^2(n)] = E[(x(n) - \sum pi{=}1\ aix(n-i))^2]$

$MSE \hat{=} E[(x(n))^2 - 2x(n) \sum pi{=}1\ aix(n-i) + (\sum pi{=}1\ aix(n-i))^2]$

$= E[(x(n))^2] - 2E[x(n) \sum pi{=}1\ aix(n-i)] + E[(\sum pi{=}1\ aix(n-i))^2]$

$= E[(x(n))^2] - 2 \sum pi{=}1\ aiE[x(n)x(n-i)] + E[\sum pi{=}1 \sum pj{=}1\ aiajx(n-i)x(n-j)]$

$= Rx(0) - 2\sum pi{=}1\ aiRx(i) + \sum pi{=}1 \sum pj{=}1\ aiajRx(i-j)$

By differentiating the previous error expression with respect to each coefficient ai of the prediction filter and setting the derivative equal to zero, we derive a set of linear equations for the filter coefficients, namely:

$\partial MSE/\partial ai = 0 \Longrightarrow \sum pk=1\ aiRx(i-j) = Rx(i),\ 1 \le i, j \le p$

This can be written in matrix form as:

$\boldsymbol{Ra} = \boldsymbol{r}$ or $\boldsymbol{a} = \boldsymbol{R}^{-1}\boldsymbol{r}$

Where:

- $\boldsymbol{R}$ is the autocorrelation matrix of dimension $p \times p$, where the (i, j) element is $Rx(i-j)$.
- $\boldsymbol{r}$ is the autocorrelation vector of dimension $p \times 1$, where the ith element is $Rx(i)$.
- $\boldsymbol{a}$ is the vector of dimension $p \times 1$ containing the prediction filter coefficients.

The autocorrelation function Rx of the random process $x(n)$ can be estimated statistically for an input sequence $x(n)$ of length N using the following relations:

$Rx(i) = 1/(N-p) \sum Nn=p+1\ x(n)x(n-i),\ 1 \le i \le p$
$Rx(i-j) = 1/(N-p) \sum Nn=p+1\ x(n-j)x(n-i),\ 1 \le i, j \le p$

The calculation of the prediction filter is done at the transmitter, and the filter coefficients are then quantized and sent to the receiver. At this point, it is worth noting that the transmitter should also use the quantized values of the filter coefficients to ensure that the transmitter and receiver function in harmony. To compute the quantized values of the coefficients, use the uniform quantizer that you will construct, setting N = 8 bits and a dynamic range of [-2, 2].

4. Uniform quantizer

In addition, you are required to implement a uniform quantizer with N binary digits, i.e., a $2N$-level quantizer that will quantize the prediction error, which has a smaller dynamic range compared to the input signal. The quantizer will specifically quantize each prediction error sample separately and will be implemented as a MATLAB function.

$\boldsymbol{y(n)} = \boldsymbol{my\_quantizer(y(n), N, min\_value, max\_value)}$

Where:

- $y(n)$: the current sample of the prediction error is the input to the quantizer
- N: the number of binary digits to be used
- max_value: the maximum acceptable value of the prediction error
- min_value: the minimum acceptable value of the prediction error
- y^(n): the quantized sample of the current prediction error sample

The quantization levels are represented by integers 1, 2, ..., $2N$, where the largest positive quantization level corresponds to the integer 1. These integers can be represented in binary using N binary digits.
centers: a vector containing the centers of the quantization regions

Specifically, the quantizer should limit the dynamic range of the prediction error to the values $[min\_value : max\_value]$, setting samples that fall outside the dynamic range to the corresponding extreme acceptable value. The quantizer then computes the quantization step Δ, the centers of each region, and determines which region the input sample belongs to, outputting the quantized sample y^(n). This sample will be used as an index for the vector centers so we can get the quantized sample as $centers(y(n))$.

The output sample of the quantizer will take values between 1 and $2N$, corresponding to the number of quantization regions. The quantized version of the output sample will take the value of the center of the quantization region to which the current input sample belongs.

Example of quantization areas for N=2 bits can be seen in the following picture:

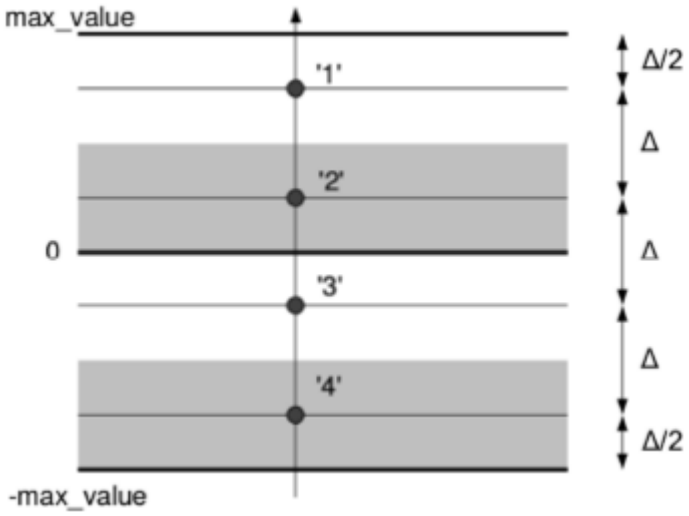

Figure 2.

5. Data source

The source you are asked to encode/decode is a signal consisting of 20,000 samples. The source samples with which you will experiment exhibit satisfactory "predictability," meaning that a current sample can be predicted (in a statistical sense) with a small prediction error by combining previous values of the same signal. The samples of the source you will experiment with are stored in a file named "source.mat." To retrieve the source data, simply type:

>> load source.mat

Questions – Part A

In the experiments you will perform, the dynamic range of the quantizer should be between the values
max_value = 3.5, min_value = -3.5.

1. Implement the above DPCM encoding/decoding system.
2. Select two values of $p \geq 5$, and for N = 1, 2, 3 bits, plot on the same graph the original signal and the prediction error y. Comment on the results. What do you observe?
3. Evaluate its performance with a graph showing the mean square prediction error as a function of N and for various values of p. Specifically, for binary digits N = 1, 2, 3, which the uniform quantizer uses for encoding the prediction signal, and for predictor orders p = 5, 10. Additionally, for each p, record and comment on the predictor coefficients in your report.
4. For N = 1, 2, 3 bits, depict the original and the reconstructed signal at the receiver for the p = 5, 10 values you selected, and comment on the reconstruction results with respect to the quantization bits.

Divide the input source samples into equal-sized, non-overlapping subsets of 5000 samples. Repeat the above questions for each subset. Comment on the results you obtained for the 5000-sample subsets and compare them with those of the 20,000 samples. What do you observe?