6 – IMAGE COMPRESSION (A)

Image compression is the science of effectively coding digital images to reduce the number of bits required in representing an image. The purpose of doing so is to reduce the storage and transmission costs while maintaining good quality.

Consider the following storage and transmission requirements:

		Storage	Transmission	
			@50 kbits/s	@512 kbits/s
Monochrome image	512×512 8 bits/pixel	256 KB	42 s	4 s
Colour image	512×512 24 bits/pixel	768 KB	126 s	12 s
Video clip	320×240 8 bits/pixel 30 frames/s 1 minute	131 MB	6.1 hrs	36 mins

It is obvious that even with broadband connections, compression is needed to deliver multimedia data in a timely manner.

Applications requiring image compression:

- facsimile transmission of graphic documents over telephone lines
- archival storage
- broadcast television (HDTV)
- digital video disk
- video conferencing
- multimedia images/video

Example

Original Lena

5:1 compression

23:1 compression

FUNDAMENTALS

Various amounts of data may be used to represent the same amount of information. Let n_1 and n_2 denote the number of information-carrying units in two data sets $\{D_1\}$ and $\{D_2\}$ that represent the same information.

$$\begin{array}{cc} \{D_1\} & \{D_2\} \\ n_1 & n_2 \end{array}$$

The *relative data redundancy* R_D of the first data set can be defined as

$$\begin{aligned} R_D &= (n_1 - n_2)/n_1 && (1) \\ &= 1 - (n_2/n_1) && (2) \\ &= 1 - \frac{1}{C_R} && (3) \end{aligned}$$

where C_R, the compression ratio, is

$$C_R = \frac{n_1}{n_2} \tag{4}$$

(a) $n_2 = n_1 \Rightarrow C_R = 1, R_D = 0$. The first representation contains no redundant data.

(b) $n_2 \ll n_1 \Rightarrow C_R \to \infty, R_D \to 1$. There is significant compression and highly redundant data.

(c) $n_2 \gg n_1 \Rightarrow C_R \to 0, R_D \to -\infty$. This is the normally undesirable case of data expansion.

n_1	n_2	R_D	C_R
$10,000$	$10,000$	0	1
$10,000$	$5,000$	0.50	2
$10,000$	$20,000$	-1	0.5

In digital image compression, three basic data redundancies can be identified and exploited:

- coding redundancy
- interpixel redundancy
- psychovisual redundancy

Coding

A code is a system of symbols (letters, bits, etc.) used to represent a body of information or a set of events. Each piece of information or event is assigned a sequence of *code symbols* called a *code word*. The number of symbols in each code word is its *length*.

Example

Consider the binary coding of the decimal digits. The correspondence of binary sequences to decimal digits given in the table is an example of a code. The 10 decimal digits are called the *message symbols*, and the 10 binary sequences are the *code words*.

Decimal digit	*Binary representation*
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Decoding, the inverse of coding, reconstructs the data. In this process, one needs to know both the specific precepts according to which a data item is translated into a single code word (the "code book") and the organization of these data. In the case of a single digital image, for example, a knowledge of the number of rows and columns is required.

Example - Morse Code

Code symbols: • (dot) — (dash) (space)

Message (source) symbol	Code word	Length
a	• —	2
b	— • • •	4
c	— • —	3
d	— ••	3
e	•	1
f	•• — •	4
etc.		

A code word consists of a sequence of code symbols.

A message is a sequence of message symbols.

message	coded message
b e d	— • • • • — ••

Decoding is not always straightforward. Consider this code

Message symbol	*Code word*
a_1	0
a_2	01
a_3	001
a_4	111

This binary sequence

111001

might have arisen from

$a_4 a_3$

or from

$a_4 a_1 a_2$

A natural (or straight) binary code is one in which each event or piece of information to be encoded (such as gray-level values) is assigned one of 2^m m-bit binary code words from an m-bit binary counting sequence.

Consider the case $m = 2$. This gives us four 2-bit binary code words which we assign to four symbols:

Symbol	*Code word*
a_1	00
a_2	01
a_3	10
a_4	11

The images that we have considered so far have been coded and saved in the natural code, i.e., a gray level of value g is coded with its 8-bit binary equivalent. A gray level of 200, for example, is coded as 11001000.

Example

The table lists three codings of the letters I to P. Clearly, Code 1 is not suitable for transmission of more than one code word, i.e., one letter. For example, the sequence 1000110 may be decoded in several ways, one of which is KIIJK.

The sequence will not occur with Code 2. In this code, any sequence with a number of bits equal to a multiple of 3 is allowed, and any such sequence, e.g. 010110100, will be unambiguously decipherable. Code 3 has the same property.

Letter	*Code 1*	*Code 2*	*Code 3*
I	0	000	01
J	1	001	00
K	10	010	10
L	11	011	1101
M	100	100	1100
N	101	101	11100
O	110	110	11101
P	111	111	11110

Coding Redundancy

Assume that a discrete random variable r_k in the interval $[0, 1]$ represents the gray levels of an image, and that each r_k occurs with probability $p_r(r_k)$:

$$p_r(r_k) = n_k/n \qquad k = 0, 1, 2, \ldots, L-1.$$

L is the number of gray levels, n_k is the number of times that the kth gray level appears in the image, and n is the total number of pixels in the image

If the number of bits used to represent each value of r_k is $l(r_k)$, the average number of bits required to represent each pixel is

$$L_{avg} = \sum_{k=0}^{L-1} l(r_k)p_r(r_k) \tag{5}$$

The total number of bits required to code an $M \times N$ image is MNL_{avg}.

Example

An 8-level image has the following gray-level distribution:

r_k	$p_r(r_k)$	Code 1	$l_1(r_k)$	Code 2	$l_2(n_k)$
$r_0 = 0$	0.19	000	3	11	2
$r_1 = 1/7$	0.25	001	3	01	2
$r_2 = 2/7$	0.21	010	3	10	2
$r_3 = 3/7$	0.16	011	3	001	3
$r_4 = 4/7$	0.08	100	3	0001	4
$r_5 = 5/7$	0.06	101	3	00001	5
$r_6 = 6/7$	0.03	110	3	000001	6
$r_7 = 1$	0.02	111	3	000000	6

If a natural 3-bit binary code (Code 1) is used to represent the 8 possible gray levels,

$$L_{avg} = 3 \text{ bits}$$

.

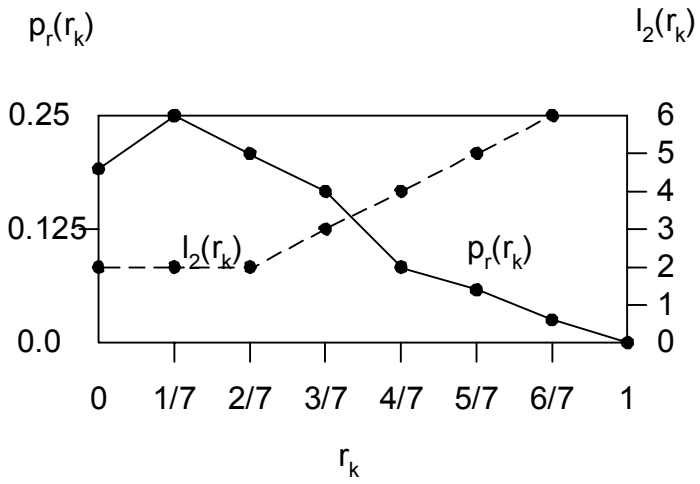
If Code 2 is used,

$$\begin{aligned} L_{avg} &= \sum_{k=0}^{7} l_2(r_k) p_r(r_k) \\ &= 2(0.19) + 2(0.25) + 2(0.21) + 3(0.16) + \\ &\quad 4(0.08) + 5(0.06) + 6(0.03) + 6(0.02) \\ &= 2.7 \text{ bits} \end{aligned}$$

Comparing Code 1 and Code 2,

$$\begin{aligned} &\text{Compression ratio} \quad C_R = 3/2.7 = 1.11 \\ &\text{Redundancy} \qquad\quad R_D = 1 - \tfrac{1}{1.11} = 0.099 \end{aligned}$$

Here we show both the histogram of the image ($p_r(r_k)$) and $l_2(r_k)$. Since $l_2(r_k)$ increases as $p_r(r_k)$ decreases, the shortest codewords in Code 2 are assigned to the gray levels that occur most frequently.



In the example, assigning fewer bits to the more probable gray levels than to the less probable ones achieves data compression. The process is known as *variable-length coding.*

If the coding is such that more code symbols than absolutely necessary are are used, we then have *coding redundancy.* In general, coding redundancy is present when the codes assigned to a set of events (such as gray-level values) have not been selected to take full advantage of the probabilities of the events. It is almost always present when an image's gray levels are represented with a straight or natural binary code.

Interpixel redundancy

This is also known as spatial redundancy, geometric redundancy, and interframe redundancy.

This arises because the value of any given pixel can be reasonably predicted from the value of its neighbours. The information carried by individual pixels is relatively small. Much of the visual contribution of a single pixel is redundant; it could have been estimated on the basis of the neighbour's values. For video images, this concept can be extended to include redundancy between frames of image data.

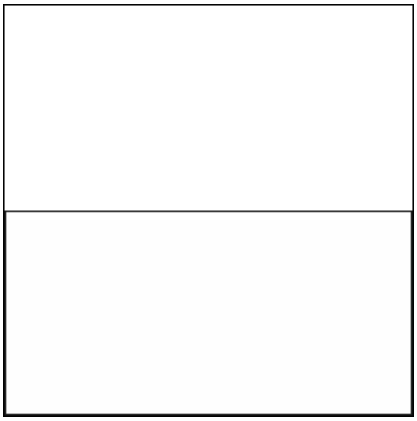
[**Example**]

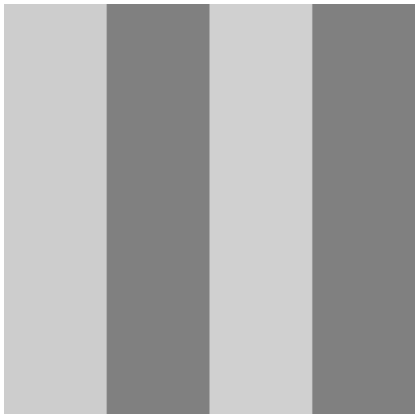
Psychovisual Redundancy

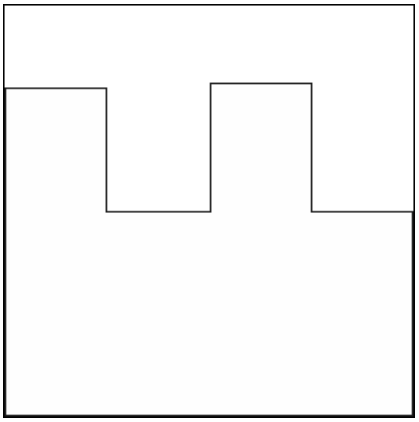
The eye does not respond with equal sensitivity to all information. Certain information simply has less relative importance than other information in normal visual processing. This information is said to be psychovisually redundant. It can be eliminated without significantly impairing the quality of image perception.

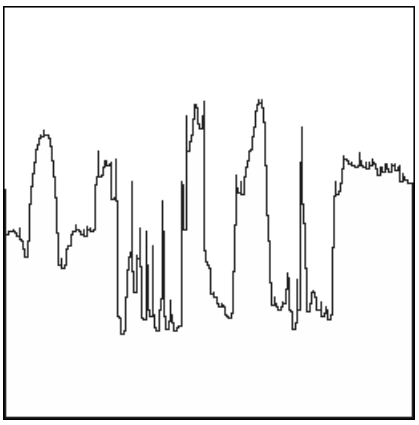
In general, an observer searches for distinguishing features such as edges of textural regions and mentally combines them into recognizable groupings. The brain then correlates these groupings with prior knowledge in order to complete hte interpretation process.

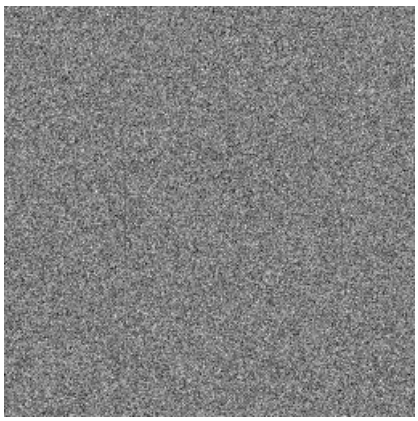
Since the elimination of psychovisually redundant data results in a loss of quantitative information, it is commonly referred to as *quantization*. This terminology is consistent with normal usage of the word, which generally means the mapping of a broad range of input values to a limited number of output values. Since it is an irreversible operation, quantization results in lossy data compression.

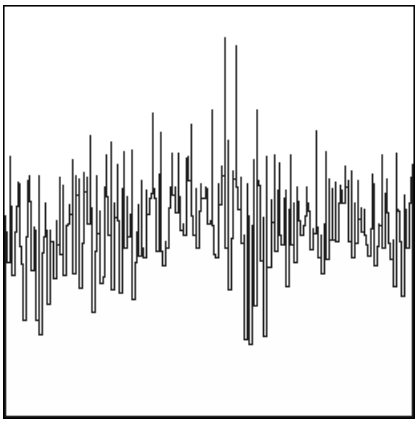












Fidelity Criteria

Compressed images may suffer from a loss of information. Two general classes of criteria are used to quantify the nature and extent of information loss: (1) objective fidelity criteria and (2) subjective fidelity criteria.

When the level of information loss can be expressed as a function of the original or input image, it is said to be based on an *objective fidelity criterion.*

Let $f(x, y)$ represent an input image and let $\hat{f}(y)$ denote an estimate or approximation of $f(x, y)$ that results from compressing and subsequently decompressing the input. For any value of x, y, the error $e(x, y)$ between $f(x, y)$ and $\hat{f}(x, y)$ can be defined as

$$e(x, y) = \hat{f}(x, y) - f(x, y) \tag{6}$$

so that the total error between the two image is

$$\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x, y) - f(x, y)] \tag{7}$$

where the images are of size $M \times N$. The *root-mean-square error*, e_{rms}, between $f(x, y)$ and $\hat{f}(x, y)$ is then the square root of the squared error averaged over the $M \times N$ array, or

$$e_{rms} = \left[\frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x, y) - f(x, y)]^2 \right]^{1/2} \tag{8}$$

A closely related objective fidelity criterion is the mean-square signal-to-noise-ratio of the compressed-decompressed image given by

$$\text{SNR}_{ms} = \frac{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \hat{f}(x, y)^2}{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x, y) - f(x, y)]^2} \tag{9}$$

In dB, we have

$$\text{SNR}_{ms} = 10 \log_{10} \left(\frac{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \hat{f}(x, y)^2}{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x, y) - f(x, y)]^2} \right) \tag{10}$$

The rms value of the signal-to-noise-ratio, denoted by SNR_{rms}, is obtained by taking the square root of SNR_{ms}.

Measuring image quality by the subjective evaluations of a human observer is often more appropriate since most decompressed images are ultimately viewed by human beings. This can be accomplished by showing a decompressed image to a viewers and averaging their evaluations. An example of a rating scale is shown in the table. The evaluations are said to be based on *subjective fidelity criteria*.

Value	*Rating*	*Description*
1	Excellent	An image of extremely high quality, as good as you could desir
2	Fine	An image of high quality, providing enjoyable viewing. Interference is not objectionable.
3	Passable	An image of acceptable quality. Interference is not objectionabl
4	Marginal	An image of poor quality; you wish you could improve it. Interference is somewhat objectionable.
5	Inferior	A very poor image, but you could watch it. Objectionable interference is definitely present.
6	Unusable	An image so bad that you could not watch it.

IMAGE COMPRESSION SYSTEMS

A compression system consists of two distinct stuctural blocks: an *encoder* and a *decoder*. An input image $f(x, y)$ is fed into the encoder, which creates a set of symbols from the input data.

After transmission over the channel, the encoded representation is fed to the decoder, where a reconstructed output image $\hat{f}(x, y)$ is generated. In general, $\hat{f}(x, y)$ may or may not be an exact replica of $f(x, y)$.

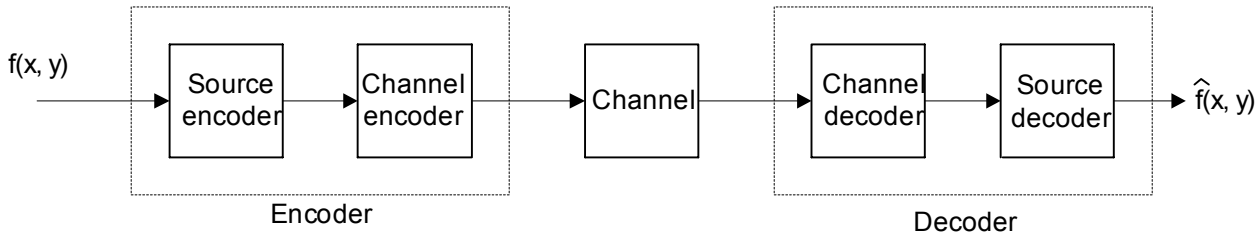
The encoder is made up of

- a *source encoder*, which removes input redundancies, and
- a *channel encoder*, which increases the noise immunity of the source encoder's output.

The decoder includes

- a *channel decoder* followed by
- a *source decoder*.

If the channel between the encoder and decoder is noise free, the channel encoder and decoder are omitted.



General model of a compression system

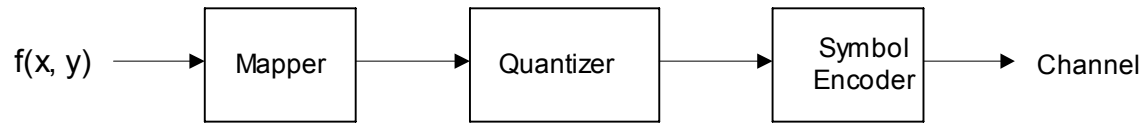
Source Encoder and Decoder

The source encoder is responsible for reducing or eliminating any coding, interpixel, or psychovisual redundancies in the input image. The approach can be modelled by a series of three independent operations.

(a) *Mapper*
This transforms the input data into a format designed to reduce interpixel redundancies in the input image. This operation generally is reversible and may or may not reduce directly the amount of data required to represent the image. Examples of such operations are run-length coding and transform coding.

(b) *Quantizer*
This reduces the accuracy of the mapper's output in accordance with some pre-established fidelity criterion. This stage reduces the psychovisual redundancies of the input image. The operation is irreversible and must be omitted when error-free compression is desired.

(c) *Symbol coder*
This stage creates a fixed- or variable-length code to represent the quantizer output.



Source encoder

All three stages are not necessarily included in every compression system. In addition, some compression techniques normally are modelled by merging blocks that are physically separate in the figure. In predictive compression, the mapper and quantizer are often represented by a single block, which simultaneously performs both operations.

The source decoder contains only two components: a *symbol decoder* and an *inverse mapper*. Because quantization results in irreversible information loss, an inverse quantizer block is not included in the general source decoder model.

Source decoder

The Channel Encoder and Decoder

The channel encoder and decoder are designed to reduce the impact of channel noise by inserting a controlled form of redundancy into the source encoded data. One of the most useful channel encoding techniques is the Hamming code.

ELEMENTS OF INFORMATION THEORY

Information theory provides the mathematical framework needed to questions such as these:

- How few data actually are needed to represent the image?
- Equivalently, is there a minimum amount of data that is sufficient to describe completely the image without loss of information?
- How efficient is one coding scheme compared to another?

Measuring Information

The fundamental premise of information theory is that the generation of information can be modelled as a probabilistic process that can be measured in a manner that agree with intuition.

A random event E that occurs with probability $P(E)$ is said to contain

$$I(E) = \log_r \frac{1}{P(E)} = -\log_r P(E) \tag{11}$$

units of information. Note that if $P(E) = 1$, (i.e., the event always occurs), $I(E) = 0$ and no information is attributed to it. Because no uncertainty is associated with the event, no information would be transferred by communicating that the event has occurred.

Consider an event E. If we are told that event E has occurred, then we have received $I(E)$ units of information, where

$$I(E) = \log_r \frac{1}{P(E)}$$

If base 2 logarithm is used ($r = 2$), the resulting unit of information is called a bit:

$$I(E) = \log_2 \frac{1}{P(E)} \text{ bits}$$

If base 10 logarithm is used, the unit of information is the Hartley:

$$I(E) = \log_{10} \frac{1}{P(E)} \text{ Hartleys}$$

We see that

$$1 \text{ Hartley} = 3.32 \text{ bits}$$

In general, if we use a logarithm to base r,

$$I(E) = \log_r \frac{1}{P(E)} \ r\text{-ary units}$$

Note that, if $P(E) = \frac{1}{2}$, $I(E) = -\log_2(\frac{1}{2})$, or 1 bit. That is, one bit is the amount of information we obtain when one of two possible equally likely alternatives is specified.

$P(E)$	$I(E)$
1	0 bit
1/2	1.0 bit
1/3	1.6 bits
1/4	2.0 bits
1/5	2.3 bits

Consider a 256-level 512×512 image. There are $256^{512 \times 512}$ different possible images. If each of these images is equally likely, the probability of a given picture is $1/256^{512 \times 512}$ and the amount of information provided by one such image is

$$\begin{aligned} I(E) &= 512 \times 512 \log_2 256 \\ &= 2.1 \times 10^6 \text{ bits} \end{aligned}$$

Entropy

Assume that an information source generates a random sequence of symbols from a finite set of possible symbols, i.e., the output of the source is a discrete random variable. The set of source symbols $\{a_1, a_2, \ldots, a_J\}$ is referred to as the *source alphabet* A. The elements of the set, denoted a_j, are called *symbols*.

The probability of the event that the source will produce symbol a_j is $P(a_j)$, and

$$\sum_{j=1}^{J} P(a_j) = 1 \tag{12}$$

The $J \times 1$ vector

$$\mathbf{z} = [P(a_1), P(a_2), \ldots, P(a_J)]^T$$

represents the set of all source symbol probabilities

$$\{P(a_1), P(a_2), \ldots, P(a_J)\}$$

The finite *ensemble* $(A, \mathbf{z})$ describes the information source completely.

The probability that the discrete source will emit symbol a_j is $P(a_j)$, so the information generated by the production of a single source symbol is (Eq. (11))

$$I(a_j) = -\log P(a_j)$$

If k source symbols are generated, then on average, symbol a_j will be output $kP(a_j)$ times. Thus the average information obtained from k outputs is

$$-kP(a_1)\,\log P(a_1) - kP(a_2)\,\log P(a_1) - \ldots - kP(a_J)\,\log P(a_J)$$

or

$$-k \sum_{j=1}^{J} P(a_j)\,\log P(a_j)$$

The average information per source output, denoted $H(\mathbf{z})$, is

$$H(\mathbf{z}) = -\sum_{j=1}^{J} P(a_j)\,\log P(a_j) \tag{13}$$

is called the *uncertainty* or *entropy* of the source. It defines the average amount of information (in r-rary units per symbol) obtained by observing a single source output. As its magnitude increases, more uncertainty and thus more information is associated with the source. If the source symbols are equally probable, the entropy of Eq. (13) is maximized and the source provides the greatest possible average information per source symbol.

Example

Consider the source $A = \{a_1, a_2, a_3\}$ with $P(a_1) = \frac{1}{2}$, $P(a_2) = P(a_3) = \frac{1}{4}$. Then

$$\begin{aligned} H(\mathbf{z}) &= \frac{1}{2}\log 2 + \frac{1}{4}\log 4 + \frac{1}{4}\log 4 \\ &= 1.5 \quad \text{bits} \end{aligned}$$

If $P(a_1) = P(a_2) = P(a_3) = \frac{1}{3}$, then

$$\begin{aligned} H(\mathbf{z}) &= \frac{1}{3}\log 3 + \frac{1}{3}\log 3 + \frac{1}{3}\log 3 \\ &= 1.58 \quad \text{bits} \end{aligned}$$

Example

Consider the English language. If the probabilities were equal for each letter, then

$$H(\mathbf{z}) = \log_2 26 = 4.64 \text{ bits}$$

i.e., the average information per letter would be 4.64 bits.

However, the relative frequencies of occurrence are unequal — e.g., e: 0.131, t: 0.105, z: 0.00077 — which leads to a reduction in average information to 4.15 bits per letter.

Example

In images, pixel values correspond to symbols. Consider the four 256×256 images, A, B, C, and D (and their respective histograms) below.

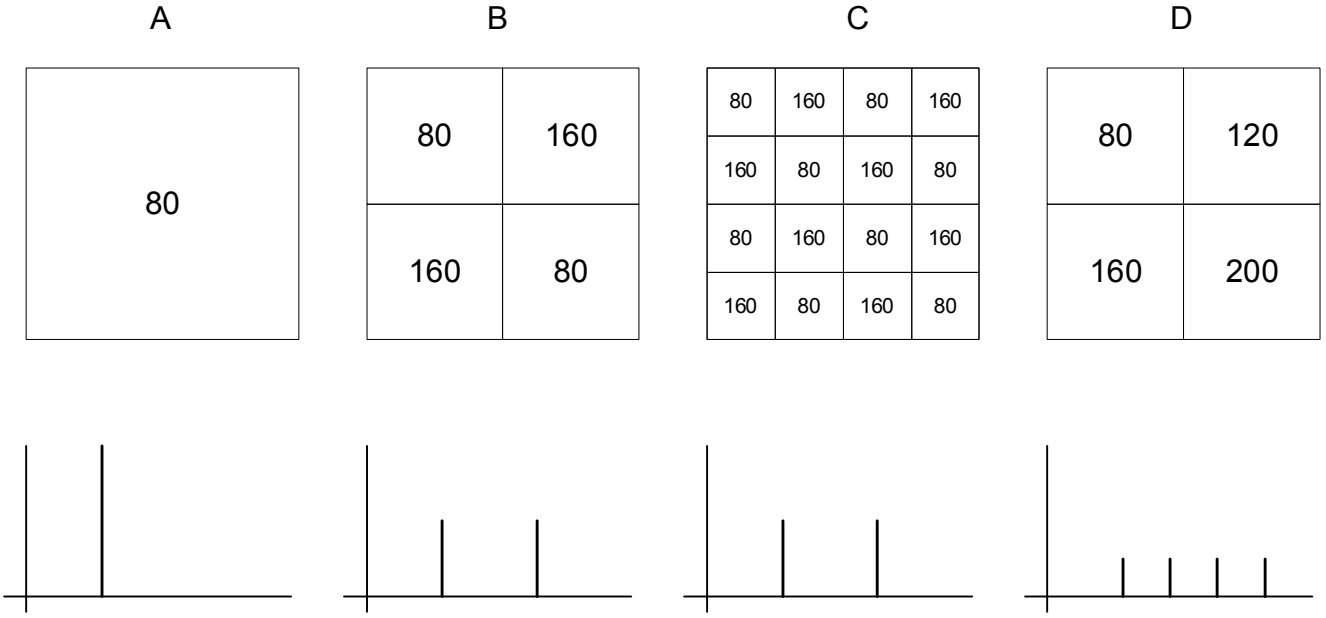


Image A:

$$H(\mathbf{z}) = -1 \log(1) = 0 \text{ bit}$$

Image B:

$$\begin{aligned} H(\mathbf{z}) &= \frac{1}{2}\log(2) + \frac{1}{2}\log(2) \\ &= 1.0 \text{ bit} \end{aligned}$$

Image C:

$$\begin{aligned} H(\mathbf{z}) &= \frac{1}{2}\log(2) + \frac{1}{2}\log(2) \\ &= 1.0 \text{ bit} \end{aligned}$$

Image D:

$$\begin{aligned} H(\mathbf{z}) &= \frac{1}{4}\log(4) + \frac{1}{4}\log(4) + \frac{1}{4}\log(4) + \frac{1}{4}\log(4) \\ &= 2.0 \text{ bits} \end{aligned}$$

The noiseless coding theorem

The *noiseless coding theorem*, also called *Shannon's first theorem*, defines the minimum average code word length per source symbol that can be achieved.

A source of information with finite ensemble $(A, \mathbf{z})$ and statistically independent symbols is called a *zero-memory* source. It is often useful to deal with *blocks* of symbols rather than individual symbols.

For example, consider a binary source emitting a sequence of 0's and 1's:

$$0\ 1\ 0\ 1\ 1\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 1\ 0$$

We may think of the bits from the source as being emitted in groups of two:

$$01\ 01\ 10\ 00\ 11\ 11\ 00\ 10$$

The binary source considered in this manner is clearly equivalent to a source with four possible symbols:

$$00,\ 01,\ 10,\ 11$$

This idea can be extended further. Suppose this binary source is considered as emitting bits in groups of three. Since there are eight possible binary sequences of length 3 (000, 001, 010, etc), then the binary source considered in this manner is equivalent to a source with a source alphabet of eight symbols.

In general, if we have a zero-memory source A with source alphabet $\{a_1, a_2, \ldots, a_J\}$, we may consider the outputs of A to be taken n at a time. Such a source is termed the the nth extension of the single-symbol source. It takes on one of J^n possible values, denoted α_i, from the set of all possible n element sequences

$$A' = \{\alpha_1, \alpha_2, \ldots, \alpha_{J^n}\}$$

where each α_i is composed of n symbols from A.

The probability of a given α_i is $P(\alpha_i)$ and is related to the single-symbol probabilities

$$P(\alpha_i) = P(a_{i_1})P(a_{i_2}) \dots P(a_{i_n}) \tag{14}$$

where α_i corresponds to $(a_{i_1}, a_{i_2}, \dots, a_{i_n})$.

The vector $\mathbf{z}'$ denotes the set of all source probabilites

$$\{P(\alpha_1), P(\alpha_2), \dots, P(\alpha_{J^n})\}$$

and the entropy of the source is

$$H(\mathbf{z}') = -\sum_{i=1}^{J^n} P(\alpha_i) \log P(\alpha_i) \tag{15}$$

It can be shown that

$$H(\mathbf{z}') = nH(\mathbf{z}) \tag{16}$$

that is, the entropy of the nth extension is n times the entropy of the corresponding single-symbol source.

Example

Consider the source $A = \{a_1, a_2, a_3\}$ with $P(a_1) = \frac{1}{2}$, $P(a_2) = P(a_3) = \frac{1}{4}$. The second extension of this source A' has nine symbols:

Symbols of A'	α_1	α_2	α_3	α_4	α_5	α_6	α_7	α_8	α_9
Sequence of A symbols	a_1a_1	a_1a_2	a_1a_3	a_2a_1	a_2a_2	a_2a_3	a_3a_1	a_3a_2	a_3a_3
Probability $P(\alpha_i)$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{16}$

$$\begin{aligned} H(\mathbf{z}') &= -\sum_{A'} P(\alpha_i) \log P(\alpha_i) \\ &= \frac{1}{4}\log 4 + 4 \times \frac{1}{8}\log 8 + 4 \times \frac{1}{16}\log 16 \\ &= 3 \text{ bits/symbol} \end{aligned}$$

(The entropy of source A is 1.5 bits, as calculated previously.)

Let L'_{avg} represent the average word length of the code corresponding to the nth extension of the nonextended source. It can be shown that

$$H(\mathbf{z}) \leq \frac{L'_{avg}}{n} < H(\mathbf{z}) + \frac{1}{n} \tag{17}$$

which, in the limiting case, becomes

$$\lim_{n\to\infty} \left[\frac{L'_{avg}}{n} \right] = H(\mathbf{z}) \tag{18}$$

Eq. (17) states Shannon's first theorem for a zero-memory source. It reveals that it is possible to make L'_{avg}/n arbitrarily close to $H(\mathbf{z})$ by coding infinitely long extensions of this source. Because $H(\mathbf{z})$ is a lower bound on L'_{avg}/n (i.e., the limit of L'_{avg}/n as n becomes large in Eq. (18) is $H(\mathbf{z})$), the efficiency η of any encoding strategy can be defined as

$$\eta = n\frac{H(\mathbf{z})}{L'_{avg}} \tag{19}$$

Example

A zero-memory information souce with source alphabet

$$A = \{a_1, a_2\}$$

has symbol probabilities

$$P(a_1) = \frac{2}{3}, \qquad P(a_2) = \frac{1}{3}$$

The entropy of the source is

$$\begin{aligned} H(\mathbf{z}) &= -\sum_{j=1}^{2} P(a_i) \log_2 P(a_i) \\ &= -\frac{2}{3}\log_2 \frac{2}{3} - \frac{1}{3}\log_2 \frac{1}{3} \\ &= 0.918 \text{ bits/symbol} \end{aligned}$$

If symbols a_1 and a_2 are represented by the binary code words 0 and 1, the average word length is

$$\begin{aligned} L'_{avg} &= \frac{2}{3}(1) + \frac{1}{3}(1) \\ &= 1 \text{ bit/symbol} \end{aligned}$$

The coding efficiency is

$$\eta = n\frac{H(\mathbf{z})}{L'_{avg}} = (1)(0.918)/1 = 0.918$$

An alternative coding scheme based on the second extension of the source is shown in the table. The four block symbols are

$$\alpha_1 = a_1a_1, \quad \alpha_2 = a_1a_2, \quad \alpha_3 = a_2a_1, \quad \alpha_4 = a_2a_2,$$

Their probabilities are easily obtained, e.g.,

$$P(\alpha_1) = P(a_1)P(a_1) = \frac{4}{9}$$

The average word length is

$$\begin{aligned} L'_{avg} &= \frac{4}{9}(1) + \frac{2}{9}(2) + \frac{2}{9}(3) + \frac{1}{9}(3) \\ &= 1.89 \text{ bits/symbol} \end{aligned}$$

The entropy of the second extension is

$$\begin{aligned} H(\mathbf{z}') &= 2H(\mathbf{z}) \\ &= 1.83 \text{ bits/symbol} \end{aligned}$$

The efficiency of the second extension is

$$\eta = 1.83/1.89 = 0.97$$

The average number of code bits per source symbol is

$$1.89/2 = 0.95 \text{ bits/symbol}$$

α_i	Source symbols	$P(\alpha_i)$	Code word	Code length
α_1	a_1a_1	4/9	0	1
α_2	a_1a_2	2/9	10	2
α_3	a_2a_1	2/9	110	3
α_4	a_2a_2	1/9	111	3