CS25110

Information Theory and Coding

3 lectures

Aim: to formalise the notion of information and explain representation principles for its electronic transfer.

- What are information and data?
- How can this data be encoded for transmission?
- Why are there physical data rate limits for transmission?
- How can errors be detected / corrected?

What is information ?

- If an event is certain to occur then we gain no information if we are told that it has occurred.
- If an event is unlikely to occur then we gain a lot of information by being told that it has occurred.

So it seems reasonable that we might quantify information using an expression which involves the ***probability*** *that the event might occur.*

- The value of the information should decrease as the probability of the event increases.

Independent Events

If we have two independent events called Q and R, then the total information gained after both events I(Q) and I(R) is I(Q+R). *If Q and R are independent*

A Formula for information

$$I(Q) = \log_b\left(\frac{1}{\text{Probability}(Q)}\right)$$

Entropy

"A measure of the amount of information that is output by a source, or throughput by a channel, or received by an observer (per symbol or second)."

Dictionary of Computing, Oxford Scientific publications

The entropy of a discrete memoryless source can be regarded as the average amount of information delivered by each symbol.

- Alphabet $A=\{a_i\}$ of size n
- output X at time t
- $P(x_i)$ = Probability$\{X_i=a_i\}$

Entropy is given:

$$H(X) = \sum_{i=0}^{n-1} P(x_i)\log_b \frac{1}{P(x_i)}$$

Units of Entropy

- b = 2 called a bit
- b = e called nat
- b = 10 called hartley

Example:
base b=2
for two equally likely symbols:

$$\Pr ob(X_1) = \tfrac{1}{2}$$
$$\Pr ob(X_2) = \tfrac{1}{2}$$
$$H(X) = \tfrac{1}{2}\log_2\left(\frac{1}{\tfrac{1}{2}}\right) + \tfrac{1}{2}\log_2\left(\frac{1}{\tfrac{1}{2}}\right)$$

$$H(X) = \tfrac{1}{2}\log_2(2) + \tfrac{1}{2}\log_2(2)$$
$$H(X) = \tfrac{1}{2}\times 1 + \tfrac{1}{2}\times 1 = 1$$

Entropy (cont.)

- Similarly, 4 equally likely symbols
  $H(X)$ = 2
- 8 equally likely symbols
  $H(X)$ = 3

Note 3 symbols in sequence, each with a choice of one of two
$H(X)$ = 1+1+1 = 3

i.e.

$$H_8(X) = 3 = H_2(X) + H_2(X) + H_2(X)$$

choice of 8 equally likely symbols

choice of 2 equally likely symbols

Codes

- For example if we required a binary code to represent an alphabet with 8 symbols, what is the average length of the codewords ?

  $H_8(X) = 3$

  therefor, average length is 3 bits
- codewords (for equally likely symbols)

  S_1 000

  S_2 001

  S_3 010

  S_4 011

  S_5 100

  S_6 101

  S_7 110

  S_8 111

Codes (cont.)

what if symbols are not equally likely ?

symbol	Pr *ob*	$P.\log_2(1/p)$
S_1	$1/4$	$1/4 \times 2 = 0.5$
S_2	$3/16$	$3/16 \times \log_2(16/3) = 0.45$
S_3	$1/8$	$1/8 \times \log_2(8) = 3/8 = 0.375$
S_4	$3/32$	$3/32 \times \log_2(32/3) = 1.28$
S_5	$3/32$	$3/32 \times \log_2(32/3) = 1.28$
S_6	$3/32$	$3/32 \times \log_2(32/3) = 1.28$
S_7	$3/32$	$3/32 \times \log_2(32/3) = 1.28$
S_8	$1/16$	$1/16 \times \log_2(16) = 0.25$
	1	$H(X) = 2.86$

therefor we need less information than that provided by 3 binary digits

note:
$\log_2(2) = 1$
$\log_2(4) = 2$
$\log_2(8) = 3$

Variable length codes

Shannon code

- Arrange symbols in decreasing order of probability.
- Calculate number of bits (N_i) for each symbol from:
  $$\log_2\left(\frac{1}{P_i}\right) \le N_i < 1 + \log_2\left(\frac{1}{P_i}\right)$$
- calculate:
  $$F_i = \sum_{j=0}^{i-1} P_j$$
  for each S_i
- Use first N_i bits of F_i after binary point

Example: Shannon code

symbol	*prob*, P_i	N_i	F_i	*SymbolCode*
S_1	¼	2	0.00000	00
S_2	3⁄16	3	0.01000	010
S_3	⅛	3	0.01110	011
S_4	3⁄32	4	0.10010	1001
S_5	3⁄32	4	0.10101	1010
S_6	3⁄32	4	0.11000	1100
S_7	3⁄32	4	0.11011	1101
S_8	1⁄16	4	0.11110	1111

Average symbol length of code (bits per symbol):

$$\sum_1^8 P_i . N_i$$
$$= \tfrac{1}{2} + \tfrac{9}{16} + \tfrac{3}{8} + 4.\tfrac{12}{32} + \tfrac{4}{16}$$
$$= \tfrac{1}{2} + \tfrac{9}{16} + \tfrac{3}{8} + \tfrac{12}{8} + \tfrac{1}{4}$$
$$= \tfrac{8}{16} + \tfrac{9}{16} + \tfrac{6}{16} + \tfrac{24}{16} + \tfrac{4}{16} = \tfrac{51}{16} = 3.1875$$

Shannon-Fano Code

- Arrange in order of decreasing probability
- Split into two groups of roughly equal probability.
- Allocate ‘0’ to one half ‘1’ to other half.
- Repeat last 2 steps for groups with greater than 1 member.

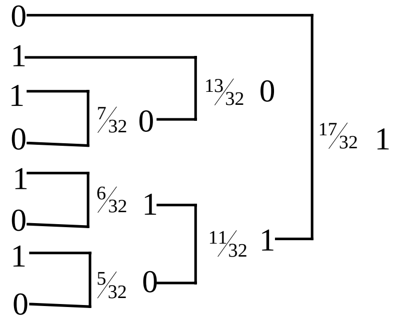
symbol	prob, P_i	Symbol Code	Code Length
S_1	$\frac{1}{4}$	00	2
S_2	$\frac{3}{16}$	010	3
S_3	$\frac{1}{8}$	011	3
S_4	$\frac{3}{32}$	1000	4
S_5	$\frac{3}{32}$	1001	4
S_6	$\frac{3}{32}$	101	3
S_7	$\frac{3}{32}$	110	3
S_8	$\frac{1}{16}$	111	3

$$\text{Average code length} = \sum_{i=1}^{8} P_i N_i = 2.9375$$

Huffman Coding

- Pick two least probable
- replace most prob. by ‘1’ and
  least prob. by ‘0’
- Add probs. and replace pair with single item with this prob.
- repeat if more than 1 item remaining
- read code from ‘top of tree’

symbol	P_i	P_i
S_1	$\frac{1}{4}$	$\frac{8}{32}$
S_2	$\frac{3}{16}$	$\frac{6}{16}$
S_3	$\frac{1}{8}$	$\frac{4}{32}$
S_4	$\frac{3}{32}$	$\frac{3}{32}$
S_5	$\frac{3}{32}$	$\frac{3}{32}$
S_6	$\frac{3}{32}$	$\frac{3}{32}$
S_7	$\frac{3}{32}$	$\frac{3}{32}$
S_8	$\frac{1}{16}$	$\frac{2}{32}$



Huffman example (cont.)

symbol	code	N_i
S_1	10	2
S_2	01	2
S_3	001	3
S_4	000	3
S_5	1111	4
S_6	1110	4
S_7	1101	4
S_8	1100	4

$$\text{Ave code length} = \sum_{i=1}^{8} P_i N_i = 2.90625$$

"Huffman coding is optimal in the sense that no other scheme uses fewer binary digits to represent a message" - Telecommunications Technology, R.L.B. Renster

A real symbol set

- Any Problems ?
  - statistics of symbol set needs to be known
  - coding scheme needs to be known at receiver
- ASCII - American Standard Code for Information Interchange
- Normally represented as 8 bits / per symbol (fixed length code)
- 7 bits for basic code
  - 8th bit used historically as parity bit
  - nowadays extra bit used for extra symbols

example ASCII code set - this slide was produced
by using the UNIX command
% man ascii

Redundancy / compression

- *Redundancy* exists when information content is less than data content.
- *Lossless* encoding or compression seeks to reduce redundancy by making the data rate closer to the information content.
- Data sources often produce lots of data.
- Sometimes in an application not all of the "information" is of interest.
- We can throw away this unwanted information to further improve transmission performance.
- These type of coding strategies are known as *lossey*.

Compression

- Lossless compression
  - fully reversible (no information lost)
  - produces more efficient data representation of information
  - applications:
    - numerical data
    - textual / ASCII data
- Lossey compression
  - Not reversible (information *is* lost)
  - based on having knowledge of data usage
  - applications:
    - images (JPEG)
    - video (MPEG, H.261)
    - sound, speech
- Most lossey compression algorithms also perform lossless compression.

Summary

- Most raw data contains some redundancy

- Techniques such as variable length codes can be used to remove this and improve channel utilisation.
-
- Some coding techniques deliberately remove some information from a data source, to gain big savings in the amount of data to transmit.

- However we may ADD redundancy to allow for detection and correction of transmission errors.