

Approximating Entropy

February 6, 2009 Harvard GR48 1

Self-Information

- If a symbol S has frequency p , its *self-information* is $H(S) = \lg(1/p) = -\lg p$.

| | | | | |
|--------|-----|-----|-----|-----|
| S | A | B | C | D |
| p | .25 | .25 | .25 | .25 |
| $H(S)$ | 2 | 2 | 2 | 2 |

| | | | | |
|--------|-----|------|------|------|
| p | .7 | .1 | .1 | .1 |
| $H(S)$ | .51 | 3.32 | 3.32 | 3.32 |

February 6, 2009 Harvard GR48 2

Self-Information = $H(S) = \lg(1/p)$

- Greater frequency \Leftrightarrow Less information
- Extreme case: $p = 1, H(S) = \lg(1) = 0$
- Why is this the right formula?
- $1/p$ is the *average length of the gaps between recurrences of S*

Average of $a, b, c, d \dots = 1/p$
 Number of bits to specify a gap is about $\lg(1/p)$

February 6, 2009 Harvard GR48 3

First-Order Entropy of Source = Average Self-Information

| | | | | | |
|------------|-----|-----|-----|-----|-----------------|
| S | A | B | C | D | $-\sum p \lg p$ |
| p | .25 | .25 | .25 | .25 | |
| $-\lg p$ | 2 | 2 | 2 | 2 | |
| $-p \lg p$ | .5 | .5 | .5 | .5 | 2 |

| | | | | | |
|------------|------|------|------|------|-------|
| p | .7 | .1 | .1 | .1 | |
| $-\lg p$ | .51 | 3.32 | 3.32 | 3.32 | |
| $-p \lg p$ | .357 | .332 | .332 | .332 | 1.353 |

February 6, 2009 Harvard GR48 4

Entropy, Compressibility, Redundancy

- Lower entropy \Leftrightarrow More redundant \Leftrightarrow More compressible
- Higher entropy \Leftrightarrow Less redundant \Leftrightarrow Less compressible
- A source of "yea"s and "nay"s takes 24 bits per symbol but contains at most one bit per symbol of information
- 010110010100010101000001 = yea
- 010011100100000110101001 = nay

February 6, 2009 Harvard GR48 5

Entropy and Compression

| | | | |
|----|----|-----|-----|
| A | B | C | D |
| .7 | .1 | .1 | .1 |
| 0 | 10 | 110 | 111 |

- No code taking only frequencies into account can be better than first-order entropy
- Average length for this code = $.7 \cdot 1 + .1 \cdot 2 + .1 \cdot 3 + .1 \cdot 3 = 1.5$
- First-order Entropy of this source = $.7 \cdot \lg(1/.7) + .1 \cdot \lg(1/.1) + .1 \cdot \lg(1/.1) + .1 \cdot \lg(1/.1) = 1.353$
- First-order Entropy of English is about 4 bits/character based on "typical" English texts

February 6, 2009 Harvard GR48 6

Second-Order Approximation to English

- Source generates all 729 *digrams* (two-letter sequences, AA ... ZZ, also A<sp>, <sp>Z, etc.) in the right proportions
- A string from a second-order source of English:
On ie antsoutinys are t inctore st be s
deamy achin d ilonasive tucoowe at
teasonare fuso tizin andy tobe seace
ctisbe

February 6, 2009

Harvard QR48

7

Second-Order Entropy

- Second-Order Entropy of a source is calculated by treating digrams as single symbols according to their frequencies
- Occurrences of q and u are not independent so it is helpful to treat qu as one
- Second-order entropy of English is about 3.3 bits/character

February 6, 2009

Harvard QR48

8

Third-Order Entropy

- Have trigrams in proper frequencies
- IN NO IST LAT WHEY CRATICT FROURE
BIRS GROCID PONDENOME OF DEMONSTURES
OF THE REPTAGIN IS REGOACTONA OF
CRE

February 6, 2009

Harvard QR48

9

Word Approximations to English

- Use English words in their real frequencies
- First-order word approximation: REPRESENTING
AND SPEEDILY IS AN GOOD APT OR COME
CAN DIFFERENT NATURAL HERE HE THE A
IN CAME THE TO OF TO EXPERT GRAY COME
TO FURNISHES THE LINE MESSAGE HAD BE
THESE

February 6, 2009

Harvard QR48

10

Second-Order Word Approximation

THE HEAD AND IN FRONTAL ATTACK ON
AN ENGLISH WRITER THAT THE
CHARACTER OF THIS POINT IS
THEREFORE ANOTHER METHOD FOR THE
LETTERS THAT THE TIME OF WHO EVER
TOLD THE PROBLEM FOR AN UNEXPECTED

February 6, 2009

Harvard QR48

11

What is entropy of English?

- Entropy is the "limit" of the information per symbol using single symbols, digrams, trigrams, ...
- Not really calculable because English is a finite language!
- Nonetheless it can be determined experimentally using **Shannon's game**
- Answer: a little more than 1 bit/character

February 6, 2009

Harvard QR48

12

Efficiency of a Code

| A | B | C | D |
|----|-----|-----|----|
| .7 | .1 | .1 | .1 |
| 0 | 100 | 101 | 11 |

- **Efficiency** of code for a source =
(entropy of source)/(average code length)
- Average code length = 1.5
- Assume that source generates symbols in these frequencies but otherwise randomly (first-order model)
- Entropy = 1.353
- Efficiency = $1.353/1.5 = 0.902$

February 6, 2009

Harvard QR48

13

Shannon's Remarkable 1948 paper

A Mathematical Theory of Communication

By C. E. SHANNON

The choice of a logarithmic base corresponds to the choice of a unit for measuring information. If the base 2 is used the resulting units may be called binary digits, or more briefly *bits*, a word suggested by J. W. Tukey. A device with two stable positions, such as a relay or a flip-flop circuit, can store one bit of information. N such devices can store N bits, since the total number of possible states is 2^N and $\log_2 2^N = N$. If the base 10 is used the units may be called decimal digits. Since

$$\log_2 M = \log_{10} M / \log_{10} 2 \\ = 3.32 \log_{10} M$$

to or are correlated according to some system with certain physical or conceptual entities. These semantic aspects of communication are irrelevant to the engineering problem. The significant aspect is that the actual message is one selected from a set of possible messages. The system must be designed to operate for each possible selection, not just the one which will actually be chosen since this is unknown at the time of design.

If the number of messages in the set is finite then this number or any monotonic function of this number can be regarded as a measure of the information produced when one message is chosen from the set, all choices being equally likely. As was pointed out by Hartley the most natural choice is the logarithmic function. Although this definition must be generalized considerably when we consider the influence of the statistics of the message and when we have a continuous range of messages, we will in all cases use an essentially logarithmic measure.

4

Shannon's Source Coding Theorem

- *No code can achieve efficiency greater than 1, but*
- *For any source, there are codes with efficiency as close to 1 as desired.*
- The proof does not give a method to find the best codes. It just sets a limit on how good they can be.

February 6, 2009

Harvard QR48

15

A Simple Prefix Code: Huffman Codes

- Suppose we know the symbol frequencies. We can calculate the (first-order) entropy. Can we design a code to match?
- *There is an algorithm that transforms a set of symbol frequencies into a variable-length, prefix code that achieves average code length approximately equal to the entropy.*
- David Huffman, 1951

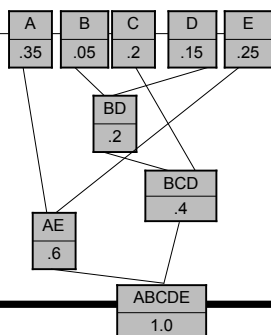


February 6, 2009

Harvard QR48

16

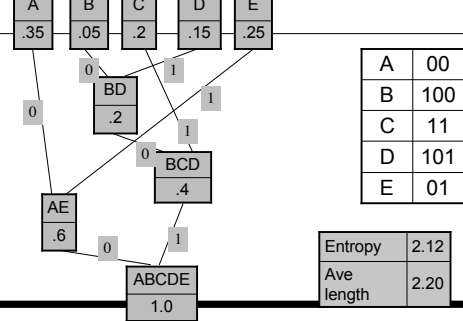
Huffman Code Example



February 6, 2009

17

Huffman Code Example



February 6, 2009

Harvard QR48

18

Efficiency of Huffman Codes

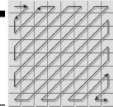
- Huffman codes are as efficient as possible *if only first-order information (symbol frequencies) is taken into account*.
- The efficiency of the Huffman code is always within 1 bit/symbol of the entropy.

February 6, 2009

Harvard GR48

19

Huffman coding used widely



- Eg JPEGs use Huffman codes to for the pixel-to-pixel **changes** in color values
 - Colors usually change gradually so there are many small numbers, 0, 1, 2, in this sequence
- JPEGs may use a fancier compression method called “arithmetic coding”
- Arithmetic coding produces 5% better compression

February 6, 2009

Harvard GR48

20

Why don't JPEGs use arithmetic coding?

- Because it is **patented by IBM**

United States Patent 4,905,297

Langdon, Jr., et al. February 27, 1990

Arithmetic coding encoder and decoder system

Abstract Apparatus and method for compressing and de-compressing binary decision data by arithmetic coding and decoding wherein the estimated probability Q_e of the less probable of the two decision events, or outcomes, adapts as decisions are successively encoded. To facilitate coding computations, an augend value A for the current number line interval is held to approximate ...

- What if Huffman had patented his code?

February 6, 2009

Harvard GR48

21

Beyond Huffman

- Sometimes it is not good enough to be within 1 bit/symbol of the entropy.
- Suppose there are only two symbols, A and B, with frequencies .99 and .01.
- Fax transmissions are like this, with A=white, B=black
- Huffman yields the code A=0, B=1, average code length = 1 bit/symbol.
- Entropy = $.99 \cdot \lg(1/.99) + .01 \cdot \lg(1/.01) = .08$
- Efficiency = $.08/1 = .08$. Need to do much better than that!

February 6, 2009

Harvard GR48

22