Bit Sequence Compression

Swanith Upadhye

(23/04/2024)

Outline

- Motivation
- Shannon’s Source Coding theorem
- Huffman Coding
- Examples
- Alternatives and Scope?

References

- Matthew V. Mahoney, “Text Compression as a Test for Artificial Intelligence”, AAAI/IAA, 1999(https://api.semanticscholar.org/CorpusID:1023392)
- https://en.wikipedia.org/wiki/Shannon%27s_source_coding_theorem
- https://en.wikipedia.org/wiki/Huffman_coding
- https://chat.openai.com/

Introduction

- “Degree” of randomness of a sequence of bits.(Disorder)

- Flip side, large compression → better info.
- Especially text compressors[1]:
  - Human text-prediction tests – 1.3 bits/char
  - Best algorithms 1.87 bits/char
- AI Test

Shannon’s Source Coding

- Lossless Compression
- *N iid random variable, each with entropy H if compressed into fewer than NH (for large N) bits guarantees loss of information.*
- Theoretical limit to lossless compressibility
- Insight to the proof : , and
- Problems arise: as discussed in class.
- Incompressability → Randomness

RLE Technique

- For a sequence:

  111110000111...

- Compressed:

  (5) x 1, (4) x 0, (3) x 1,.....

- Example:code

- Problems? Merits?

H1)Huffman Coding

- Highly compressibility
- How to?
  - Give smallest bit seq. to most frequent character
  - Prefix code to avoid overlapping messages:
- Prefixing Eg: suppose a→ 0, b→ 1, c→ 01

  Then 0101: either abab or abc or cab or cc.

H2) Example

Input (*A*, *W*)	Symbol (a_i)	a	b	c	d	e	**Sum**
	Weights (w_i)	0.10	0.15	0.30	0.16	0.29	= 1
Output *C*	Codewords (c_i)	`010`	`011`	`11`	`00`	`10`	
	Codeword length (in bits) (l_i)	3	3	2	2	2	
	Contribution to weighted path length ($l_i\, w_i$)	0.30	0.45	0.60	0.32	0.58	$L(C)$ = 2.25
Optimality	Probability budget (2^{-l_i})	1/8	1/8	1/4	1/4	1/4	= 1.00
	Information content (in bits) ($-\log_2 w_i$) ≈	3.32	2.74	1.74	2.64	1.79	
	Contribution to entropy ($-w_i \log_2 w_i$)	0.332	0.411	0.521	0.423	0.518	$H(A)$ = 2.205

- https://en.wikipedia.org/wiki/Huffman_coding

H3) Prefix tree

https://demo.tinyray.com/huffman

aaaaaaaaaabbbbbbbbbbbbbbbccccccccccccccccccccccccccccccdd

ddddddddddddddeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee

H4)Huffman Bit Seq

- Problem?
- https://demo.tinyray.com/huffman
- How to approach? → Repeating Pattern naming maybe?

Closing Remarks

- Compression problem → Pattern Recognition problem(AI)
  - Eg: Arithmetic (N – point correlator)
- Some methods - more suited to particular bit seq
- Randomness dispute: Unsettled
  - (Hidden patterns? Other techniques?)
-