CMPSCI 240: Reasoning about Uncertainty

Lecture 24: More Information Theory

Andrew McGregor

University of Massachusetts

Last Compiled: April 28, 2017

Information Theory

- Encoding messages/files as binary strings. . .
- Information Transmission: How to talk over a garbled phone line.
- Information Compression: How you'd design a language if you like to keep your conversations brief.

Outline

Encoding Messages with Redundancy: Error Correcting

- Suppose you now have 16 possible messages corresponding to

$$0000, 0001, 0010, \ldots, 1111$$

- Now consider adding 3 bits $y_1 y_2 y_3$ to each string $x_1 x_2 x_3 x_4$ where

$$\begin{aligned} y_1 &= x_1 + x_2 + x_4 \pmod 2 \\ y_2 &= x_1 + x_3 + x_4 \pmod 2 \\ y_3 &= x_2 + x_3 + x_4 \pmod 2 \end{aligned}$$

- After encoding, the messages become a set of codewords

$$0000000\ ,\ 0001111\ ,\ 0010011\ ,\ \ldots\ ,\ 1111111$$

all we could check all pairs of strings differ in at least three positions.

Coding: Minimum Distance

- A code has minimum distance δ if all pairs of different codewords differ in at least δ positions and there exists two different codewords that differ in exactly δ positions.
- Suppose codeword c is sent and $< \delta/2$ bits are flipped. Then the codeword most similar to the received string is the sent codeword:
  - Let z be the received string and let c' be the codeword that looks most similar to z. Then,

$$d(z, c') \leq d(z, c) < \delta/2$$

  - Then,

$$d(c, c') \leq d(c, z) + d(z, c') \leq 2d(z, c) < \delta$$

  and so $c = c'$ since all different codewords differ in $\geq \delta$ positions.

Outline

Encoding for Compression

- Sometimes want to send the minimum number of bits to convey our message.
- If there are k different messages that we need to send, then we know that sending $n \geq \lceil \log_2 k \rceil$ bits is necessary and sufficient.
- But if some messages are more common then other messages, maybe we can use short binary strings for common messages and longer strings for other messages. But this isn't straightforward...

Example

- Suppose we have 6 messages with different probabilities that we'll want to send the message

"hello"	:	0.3
"goodbye"	:	0.25
"elephant"	:	0.15
"dog"	:	0.13
"giraffe"	:	0.09
"hippo"	:	0.08

- Encode "hello" and "goodbye" and 0 and 1 and other messages as $00, 01, 10, 11$ respectively
- But how would you interpret 010?
- We'll solve this with Huffman encoding...

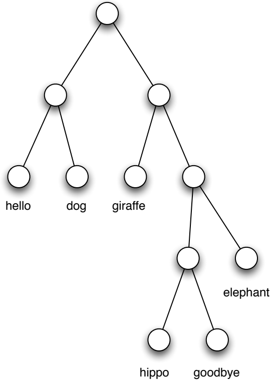
Encode the alphabet as a binary string

- If there are k messages, consider a binary tree with k leaves where each leaf is labeled by one of the messages.
- For each leaf u, consider the path from root to leaf. Associate the path with a binary string in a natural way, e.g., if the path to the leaf goes left-left-right we consider the binary string 001.
- Then the average encoding length for a message is

$$A = \sum_{u} d(u) P(u)$$

where $P(u)$ is the probability of the message associated with leaf u and $d(u)$ is the length of the path from the root to leaf u.

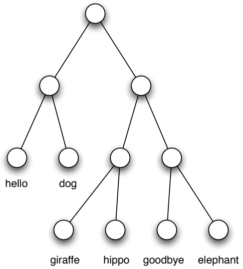
Example Encoding 1



	Probability	Encoding
“hello”	0.3	00
“goodbye”	0.25	1101
“elephant”	0.15	111
“dog”	0.13	01
“giraffe”	0.09	10
“hippo”	0.08	1100

Average length $= 0.3 \cdot 2 + 0.25 \cdot 4 + 0.15 \cdot 3 + 0.13 \cdot 2 + 0.09 \cdot 2 + 0.08 \cdot 4 = 2.81$

Example Encoding 2



	Probability	Encoding
"hello"	0.3	00
"goodbye"	0.25	110
"elephant"	0.15	111
"dog"	0.13	01
"giraffe"	0.09	100
"hippo"	0.08	101

$$\text{Average length} = 0.3\cdot2+0.25\cdot3+0.15\cdot3+0.13\cdot2+0.09\cdot3+0.08\cdot3 = 2.48$$

Huffman's Algorithm

- Find the two messages a_1 and a_2 with lowest probability and replace them by a new message b whose probability equals $P(a_1) + P(a_2)$.
- Recurse on the new set of messages and then replace the leaf $a_1 a_2$ with an internal node which has leaves a_1 and a_2.

Theorem

Huffman's Algorithm produces an encoding tree T such that no other tree has smaller average depth.

Guessing Game

- Suppose that I'm thinking of a number that is equally likely to be any of $\{1, 2, 3, \ldots, 128\}$.
- How many yes/no questions do you expect to need before you find my number? $\log_2 128 = 7$
- Suppose that I'm thinking of a number in $\{1, 2, 3, \ldots, 128\}$ but you know that I'm thinking of i with probability p_i?
- How many yes/no questions do you now expect to need?

Definition

The entropy of a set of probabilities $p_1, p_2, \ldots, p_n$ is defined as

$$H = \sum_{i=1}^{n} p_i \log \frac{1}{p_i}$$

Can be shown that you need roughly H questions in expectation.

Huffman Encoding Helps Design a Good Guessing Game

- Given the probabilities $p_1, p_2, \ldots, p_n$, design Huffman tree.
- At each step, ask which subtree my number lies in.
- Can be shown that the expected number of questions is between H and $H + 1$!

Theorem

Suppose every $p_i \in \{1/2, 1/4, 1/8, \ldots\}$. Then Huffman's Algorithm produces an encoding where the expected depth of a character is H.

Proof

- Proof by induction on the n.
- Base Case: When $n = 2$, we have $p_1 = 1/2$, $p_2 = 1/2$, and both leaves have depth 1. So expected depth is 1. This equals the entropy

$$H = \frac{1}{2} \times \log \frac{1}{1/2} + \frac{1}{2} \times \log \frac{1}{1/2} = 1 \ .$$

- Induction Hypothesis: Assume that for any distribution $q_1, \ldots, q_{n-1}$ over $n - 1$ characters, Huffman gives a tree with expected depth,

$$\sum_{i=1}^{n-1} q_i \log \frac{1}{q_i}$$

- If theorem being true with $n - 1$ characters implies theorem is true for with n characters, then it's true for any number of characters.

Induction Step

- Suppose n probabilities are $p_1 \geq \ldots \geq p_{n-1} \geq p_n$ and we merge the last two. Now have $n-1$ probabilities $p_1, p_2, \ldots, p_{n-2}, p_{n-1} + p_n$.
- Let A be the average depth of the Huffman tree, it can be shown:

$$A(p_1, \ldots, p_{n-1}, p_n) = A(p_1, \ldots, p_{n-1} + p_n) + p_{n-1} + p_n$$

- Since all probabilities are in $\{1/2, 1/4, 1/8, \ldots\}$, it can be shown:

$$H(p_1, \ldots, p_n) = H(p_1, \ldots, p_{n-1} + p_n) + p_{n-1} + p_n$$

- By induction hypothesis,

$$A(p_1, \ldots, p_{n-1} + p_n) = H(p_1, \ldots, p_{n-1} + p_n)$$

  and so,

$$A(p_1, \ldots, p_{n-1}, p_n) = H(p_1, \ldots, p_{n-1}+p_n)+p_{n-1}+p_n = H(p_1, \ldots, p_n)$$

That's All!

- No more lectures or discussion sections!
- Last homework due Tuesday and last scheduled quiz due Monday.
- Final Exam: 1pm, Tuesday 9th May, Thompson Hall 104
- Bonus Office Hours:

  3:30pm Friday 28th April and 2pm Monday 8th May

  and I'll post an extra credit revision quiz.