DCSP-3: Minimal Length Coding

Jianfeng Feng

Department of Computer Science Warwick Univ., UK

Jianfeng.feng@warwick.ac.uk

http://www.dcs.warwick.ac.uk/~feng/dcsp.html
Automatic Image Caption (better than human)

- "girl in pink dress is jumping in air."
- "black and white dog jumps over bar."
- "young girl in pink shirt is swinging on swing."
- "man in blue wetsuit is surfing on wave."
- "little girl is eating piece of cake."
- "baseball player is throwing ball in game."
- "woman is holding bunch of bananas."
- "black cat is sitting on top of suitcase."
This Week’s Summary:

get familiar with 0 and 1

• Information theory

• Huffman coding: code events as economic as possible
Information sources

- \( X = \{x_1, x_2, \ldots, x_N\} \) with a known probability
- \( P(x_i) = p_i, \; i=1,2,\ldots,N \)

Example 1:

\[
X = (x_1 = \text{lie on bed at 12 noon today})
\quad (x_2 = \text{in university at 12 noon today})
\quad (x_3 = \text{attend a lecture at 12 noon today})
\]

\( = (B, U, L) \)

\( p = (1/2,1/4,1/4), \)

- \( H(X) = .5 \times 1 + 2 \times 1/4 + 2 \times 1/4 = 1.5 \) (Entropy)
- \( B=0, \; U=1, \; L=01 \) (coding)
- \( L_s = 0.5 \times 1 + 0.25 \times 1 + 0.25 \times 2 = 1.25 \) (average coding length)
Information sources

• Example 2. Left: information source \( p(x_i), i = 1, \ldots, 27 \) 
right: codes

To be, or not to be, that is the question—Whether 'tis Nobler in the mind to suffer
The Slings and Arrows of outrageous Fortune, Or to take Arms against a Sea of troubles
And by opposing end them? To die, to sleep—

As short as possible

01110 00001111111
11111100000000000
1111000000011000
100010000010000
101111111100000
Information source coding

• Replacement of the symbols (naked run/office in PM example) with a binary representation is termed *source coding*.

• In any coding operation we replace the symbol with a codeword.

• The purpose of source coding is to reduce the number of bits required to convey the information provided by the information source:

  *minimize the average length of codes.*

• Conjecture: an information source of entropy $H$ needs on average only $H$ binary bits to represent each symbol.
An instantaneous code can be found that encodes a source of entropy $H(X)$ with an average number $L_s$ (average length) such that

$$L_s \geq H(X)$$

Shannon's first theorem
How does it work?

- Like many theorems of information theory, the theorem tells us nothing of how to find the code.
- However, it is useful results.
- Let us have a look how it works
Example

- Look at the activities of PM in three days with $P(O)=0.9$
- Calculate probability
- Assign binary codewords to these grouped outcomes.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>OOO</th>
<th>OON</th>
<th>ONO</th>
<th>NOO</th>
<th>NNO</th>
<th>NON</th>
<th>ONN</th>
<th>NNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probability</td>
<td>0.729</td>
<td>0.081</td>
<td>0.081</td>
<td>0.081</td>
<td>0.009</td>
<td>0.009</td>
<td>0.009</td>
<td>0.001</td>
</tr>
<tr>
<td>Codeword</td>
<td>0</td>
<td>1</td>
<td>01</td>
<td>10</td>
<td>11</td>
<td>00</td>
<td>000</td>
<td>111</td>
</tr>
<tr>
<td>Codeword Length (in bits)</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Weighted code length</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Entropy</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Variable length source coding
Table 1 shows such a code, and the probability of each code word occurring.

Entropy is
\[
H(X) = -0.729 \log_2(0.729) - 0.081 \log_2(0.081) \times 3 \\
- 0.009 \log_2(0.009) \times 3 - 0.001 \log_2(0.001) \\
= 1.4070
\]

The average length of coding is given by
\[
L_s = 0.729 \times 1 + 0.081 \times 1 + 2 \times 0.081 \times 2 + 2 \times 0.009 \times 2 \\
+ 3 \times 0.009 + 3 \times 0.001 \\
= 1.2
\]
Moreover, without difficulty, we have found a code that has an average bit usage less than the source entropy.
However, there is a difficulty with the code in Table 1.

Before a code word can be decoded, it must be parsed.

Parsing describes that activity of breaking the message string into its component codewords.
Example

• After parsing, each codeword can be decoded into its symbol sequence.

• An instantaneously parsable code is one that can be parsed as soon as the last bit of a codeword is received.
Instantaneous code

- An instantaneous code must satisfy the prefix condition: that no codeword may be a prefix of any other code.

- For example: in the codeword, we should not use 1 11 to code two events. When we receive 11, it could be ambiguous.

- This condition is not satisfied by the code in Table 1.
Huffman coding

• The code in Table 2, however, is an instantaneously parsable code.

• It satisfies the prefix condition.
Huffman coding

<table>
<thead>
<tr>
<th>Sequence</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probability</td>
<td>0.729</td>
<td>0.081</td>
<td>0.081</td>
<td>0.081</td>
<td>0.009</td>
<td>0.009</td>
<td>0.009</td>
<td>0.001</td>
</tr>
<tr>
<td>Codeword</td>
<td>1</td>
<td>011</td>
<td>010</td>
<td>001</td>
<td>00011</td>
<td>00010</td>
<td>00001</td>
<td>00000</td>
</tr>
<tr>
<td>Codeword Length (in bits)</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

**Entropy**

Table 2: OOO=A, OON=B, ONO=C, NOO=D, NNO=E, NON=F, ONN=G, NNN=H

\[ L_s = 0.729 \times 1 + 0.081 \times 3 \times 3 + 0.009 \times 5 \times 3 + 0.001 \times 5 \]

\[ = 1.5980 \] (remember entropy is 1.4)
Huffman coding

Decoding

1 1 1 0 1 1 0 1 0 1 1 0 0 0 0 0 0 0 0 0 1
Huffman coding

- The derivation of the Huffman code tree is shown in the following Figure and the tree itself is shown in the next Figure.

- In both these figures, the letter A to H have been used in replace of the sequence in Table 2 to make them easier to read.
Huffman coding

Creating the tree:

1. Start with as many leaves as there are symbols.
2. Queue all leaf nodes into the first queue (in order).
3. While there is more than one node in the queues:
   - Remove two nodes with the lowest weight from the queues.
   - Create a new internal node, with the two just-removed nodes as children (either node can be either child) and the sum of their weights as the new weight.
   - Update the parent links in the two just-removed nodes to point to the just-created parent node.
4. Queue the new node into the second queue.
5. The remaining node is the root node; the tree has now been generated.
Prefix condition is obviously satisfied since in the tree above, each branch codes one alphabetic.
Huffman coding

- For example, the code in Table 2 uses 1.6 bits/symbol which is only 0.2 bits/symbol more bits per sequence than the theorem tells us is the best we can do.

- We might conclude that there is little point in expending the effort in finding a code less satisfying the inequality above.
Another thought

• How much have we saved in comparison with the most naïve idea?

• i.e. O=1, N=0

• \( L_s = 3 \left[ P(OOO) + \ldots + P(NNN) \right] = 3 \), halving it
My most favourite story (History)

• In 1951, David A Huffman and his MIT information theory classmates were given the choice of a term paper or a final exam.

• The Professor, Robert M Fano, assigned a term paper on the problem of finding the most efficient binary code.

• Huffman, unable to prove any codes were the most efficient, was about to give up when he hit upon the idea of using a frequency-sorted binary tree and quickly proved this method the most efficient.

• In doing so, the student outdid his professor, who had worked with information theory inventor Clude Shannon to develop an optimal code.

• By building the tree from the bottom up instead of the top down, Huffman avoided the major flaw of the suboptimal Shannon-Fano coding.
Coding English: Huffman Coding

Frequency for alphabetics
Turbo coding

• Using Bayesian theorem to code and decode

• Bayesian theorem basically said we should employ priori knowledge as much as possible

• Read yourself