Unidade Curricular

“Informação e Codificação”

António José Ribeiro Neves

an@ua.pt

https://www.ua.pt/pt/uc/15264

Outline

- Compression
- Modeling
- Coding
- Guiding questions
- Challenges
- Exercices

- **LOSSLESS COMPRESSION**

Lossless compression techniques, as their name implies, involve no loss of information. If data have been losslessly compressed, the original data can be recovered exactly from the compressed data. Lossless compression is generally used for applications that cannot tolerate any difference between the original and reconstructed data.

- **LOSSY COMPRESSION**

Lossy compression techniques involve some loss of information, and data that have been compressed using lossy techniques generally cannot be recovered or reconstructed exactly. In return for accepting this distortion in the reconstruction, we can generally obtain much higher compression ratios than is possible with lossless compression.

- **MEASURES OF PERFORMANCE**

A compression algorithm can be evaluated in a number of different ways. We could measure the relative complexity of the algorithm, the memory required to implement the algorithm, how fast the algorithm performs on a given machine, the amount of compression, and how closely the reconstruction resem- bles the original.

Some guiding questions

What is the relationship between modeling and coding?

How do probability models relate to coding efficiency?

What is entropy, and how does it relate to coding?

What is a Markov model, and how does it help in coding?

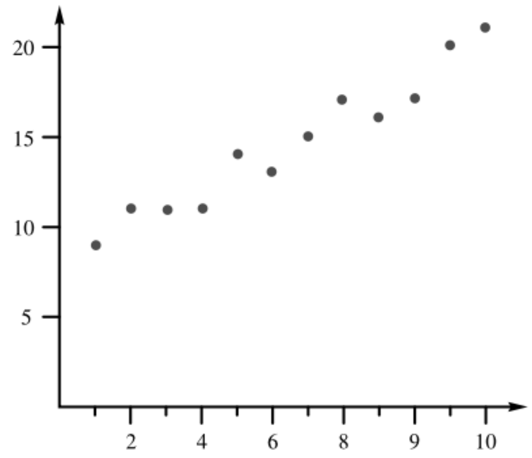
What are the trade-offs between simple probability models and more complex models like Markov models in coding?

Modeling

- In this phase, we try to extract information about any redundancy that exists in the data and describe the redundancy in the form of a model.

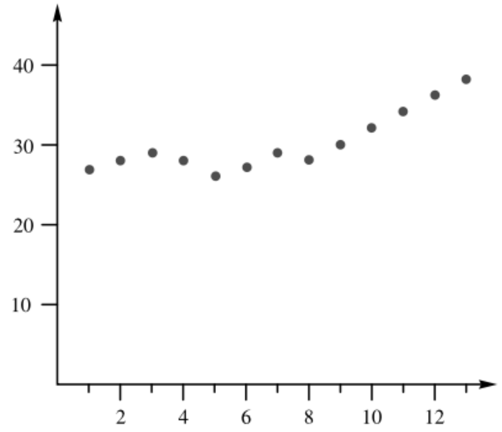
Example 1.2.1. Consider the following sequence of numbers $\{x_1, x_2, x_3, \ldots\}$:

9	11	11	11	14	13	15	17	16	17	20	21



Example 1.2.2. Consider the following sequence of numbers:

27	28	29	28	26	27	29	28	30	32	34	36	38



Coding

- Coding: a description of the model and a “description” of how the data differ from the model are encoded, generally using a binary alphabet.
- The difference between the data and the model is often referred to as the residual.
- Techniques that use the past values of a sequence to predict the current value and then encode the error in prediction, or residual, are called predictive coding schemes.
- A very different type of redundancy is statistical in nature.

Table 1.1 A Code With Codewords of Varying Length

a	1
$\cdot$	001
b	01100
f	0100
n	0111
r	000
w	01101
y	0101

Example 1.2.3. Suppose we have the following sequence:

$$a\cdot barrayaran\cdot array\cdot ran\cdot far\cdot faar\cdot faaar\cdot away$$

PROBABILITY MODELS

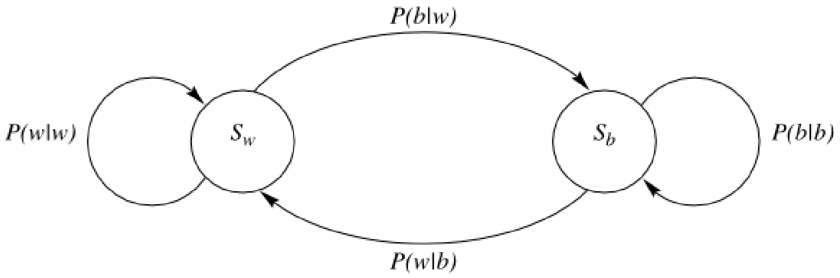
For a source that generates letters from an alphabet A = {a1, a2, . . . , aM }, we can have a probability model (keeping the independence assumption) P = {P(a1),P(a2),...,P(aM)}.

We can compute the entropy of the source using

$$H(\mathcal{S}) = -\sum P(X_1) \log P(X_1).$$

If the assumption of independence does not fit with our observation of the data, we can generally find better compression schemes if we discard this assumption.

MARKOV MODELS



For example, consider a binary image. The image has only two types of pixels, white pixels and black pixels.

We know that the appearance of a white pixel as the next observation depends, to some extent, on whether the current pixel is white or black.

Therefore, we can model the pixel process as a discrete time Markov chain.

The entropy of a finite state process with states S_i is simply the average value of the entropy at each state

$$H(S_w) = -P(b|w)\log P(b|w) - P(w|w)\log P(w|w)$$

$$H = \sum_{i=1}^{M} P(S_i)H(S_i).$$

Example

Example 2.3.1 (Markov model). To see the effect of modeling on the estimate of entropy, let us calculate the entropy for a binary image, first using a simple probability model and then using the finite state model described above. Let us assume the following values for the various probabilities:

$$P(S_w) = 30/31 \qquad P(S_b) = 1/31$$

$$P(w|w) = 0.99 \qquad P(b|w) = 0.01 \qquad P(b|b) = 0.7 \qquad P(w|b) = 0.3.$$

Example in text compression

- Markov models are particularly useful in text compression, because the next letter in a word is generally heavily influenced by the preceding letters.
- The use of Markov models for written English appeared in the original work of Shannon. In 1951, he estimated the entropy of English to be in between about 0.6 and 1.3 bits per letter.
- The frequency of letters in English is far from uniform: the most common, “E”, occurs about 13%, whereas the least common, “Q” and “Z”, occur about 0.1% of the time.
- The frequency of pairs of letters is also nonuniform. For example, the letter “Q” is always followed by a “U”. The most frequent pair is “TH” (occurs about 3.7%).
- For constructing higher-order models, the size of the model grows exponentially (What is size of the model consider a third-order Markov model?)?

Online learning

Often, the finite-context models are "learned" online, i.e., as the sequence of symbols is processed

In the case of online operation the probabilistic models are continuously adapting

A table (or some other more sophisti-cated data structure, such as an hash-table) is used to collect counts that represent the number of times that each symbol occurs in each context.

x_{i-3}	x_{i-2}	x_{i-1}	n_0	n_1
0	0	0	10	25
0	0	1	4	12
0	1	0	15	2
0	1	1	3	4
1	0	0	34	78
1	0	1	21	5
1	1	0	17	9
1	1	1	0	22

Binary source modeled by an order-3 finite-context model

More about codes...

- Dictionary compression (eg. We see redundancy in the form of words that repeat often)
- Often the structure or redundancy in the data becomes more evident when we look at groups of symbols
- Situations in which it is easier to take advantage of the structure if we decompose the data into a number of components. We can then study each component separately and use a model appropriate to that component.
- Standards allow products developed by different vendors to communicate: a number of standards for various compression applications have been approved

Project

- Use the compression utility on your computer to compress different files. Study the effect of the original file size and type on the ratio of the compressed file size to the original file size.

Some guiding questions

- What is a decodable code, and why is it important in data compression?
- What is a prefix code, and how does it ensure decodability?
- Why might we prefer prefix codes over other types of codes?
- How does the structure of a code impact its efficiency and reliability?
- Can a code be decodable but not a prefix code? If so, give an example.

Codes

- Coding means the assignment of binary sequences to elements of an alphabet
- The set of binary sequences is called a code, and the individual members of the set are called codewords
- An alphabet is a collection of symbols
- A code uses the same number of bits to represent each symbol is called a fixed-length code.
- If we want to reduce the number of bits required to represent different messages, we need to use a different number of bits to represent different symbols
- If we use fewer bits to represent symbols that occur more often, on the average we would use fewer bits per symbol.
- The average number of bits per symbol is often called the rate of the code.

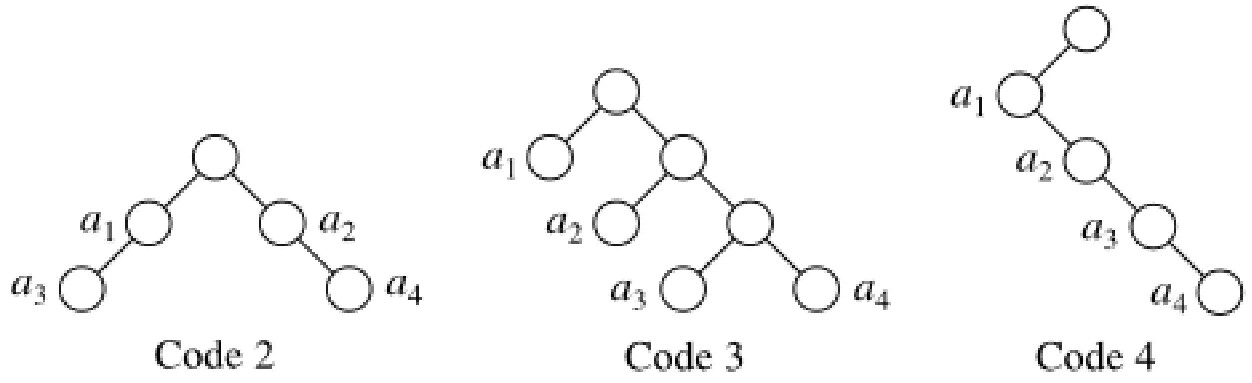
Example

Table 2.2 Four Different Codes for a Four-Letter Alphabet

Letters	Probability	Code 1	Code 2	Code 3	Code 4
a_1	0.5	0	0	0	0
a_2	0.25	0	1	10	01
a_3	0.125	1	00	110	011
a_4	0.125	10	11	111	0111
Average length		1.125	1.25	1.75	1.875

- To be useful, a code should have the ability to transfer information in an unambiguous manner
- Each symbol has to be assigned a unique codeword.
- Any given sequence of codewords can be decoded in one, and only one, way - unique decodability
- Code 3 is called an instantaneous code - Code 4 is not

Prefix Codes



- A code in which no codeword is a prefix to another codeword is called a prefix code
- Note that apart from the root node, the trees have two kinds of nodes—nodes that give rise to other nodes and nodes that do not
- The first kind of nodes are called internal nodes, and the second kind are called external nodes or leaves
- In a prefix code, the codewords are only associated with the external nodes
- A code that is not a prefix code, such as Code 4, will have codewords associated with internal nodes.