Assignment 1 - Copy models for data compression

Algorithmic Information Theory (2022/23)
Universidade de Aveiro

Martinho Tavares, 98262, martinho.tavares@ua.pt
Nuno Cunha, -, - Pedro Lima, -, -

March 28, 2023

1 Introduction

2 Work organization

3 Copy model

...

In order to evaluate whether the copy model can provide acceptable results, we can use a baseline below which we expect the model to report the file's entropy. We decided to use, as a baseline, the entropy considering each symbol's relative frequency in the entire file, which is given by:

$$H(X) = -\sum_{x \in X} p(x) \log p(x)$$

With this value in mind, we evaluated the model as a whole with different values for its parameters, on different files. The files chosen for testing are present in the repository[1], and they have the following baselines:

- `chry.txt`: ...
- ...: ...

...

Throughout this section, the different program parameters are detailed, and their effect on the model's performance is studied.

[1] `https://github.com/NMPC27/TAI-G7-Lab1`

3.1 Pattern size

When choosing a pointer in the past from which to start copying, we need to look for an occurrence of the same k-sized pattern as the one we are currently on. Thus, k is one of the parameters that affects program performance, where k is a positive integer.

On one hand, a lower value of k means there will be more occurrences of each pattern, since there will be lesser possible k-sized patterns, which will result in more oportunities to copy from the past. On the other, a larger k will result in less occurrences for each pattern, which results in more scarce predictions, and thus more overall guesses.

For a given k, the number of possible patterns M is equal to the number of permutations of the symbols of alphabet A: $M = |A|^k$. The results on changing this value will be studied taking this exponential growth into account.

3.1.1 Results

...

3.2 Smoothing parameter alpha

At each coding step, the probability of correctly predicting the current character is given by the formula:

$$P(\text{hit}) = \frac{N_h + \alpha}{N_h + N_f + 2\alpha}$$

where N_h and N_f are respectively the number of hits and misses, and α is the smoothing parameter. The smoothing parameter is used to avoid the model having a probability of 0 when there are no hits or misses, which will happen when no prediction has been made yet, and also avoid assuming with complete certainty (probability of 1) that the prediction will be correct when no fails were made yet, which will result in infinite information in case the prediction fails.

If α is lower, we expect the reported probabilites to be more "certain" with a greater possibility of reaching high values. If α is greater, the smoothing effect should be more pronounced, producing probability values that hardly increase, which should result in relatively less information when predictions fail but also lesser returns on the correct predictions.

This trade-off between α and the reported information, especially regarding the information spikes that may occur, will be studied in the results.

3.2.1 Results

...

3.3 Base probability distribution

For each symbol in the file, a probability distribution has to be generated according to the prediction probability. In case a given prediction fails, we need to distribute the complement of the probability of a correct prediction $1 - P(\text{hit})$ among the other symbols of the alphabet, except the one that we predicted would occur (which has probability $P(\text{hit})$).

The choice of distribution also influences the guesses that are made. When a prediction can't be made, the model assumes a distribution for the entirety of the alphabet, which is the same probability distribution as the use used when distributing the complement of the prediction probability.

We developed two simple approaches to distributing the probability: a uniform distribution and a frequency distribution.

The uniform distribution assigns to each symbol a probability equal to:

$$\frac{1 - P(\text{hit})}{|A| - 1}$$

On the other hand, the frequency distribution assigns to each symbol $a \in A$ a probability equal to:

$$(1 - P(\text{hit})) \times p(a)$$

where $p(a)$ is the probability of a in the whole file.

Both distributions are compared especially in their effect on the overall reported information, when k is large and thus more guesses are made, as well as when α is large and the prediction fails should report. We assume that, since the frequency distribution models the overall distribution of the symbols, it should provide better results. We should note, however, that the file's content may have locality, i.e. select portions in the file where the symbol distribution is skewed. Therefore, we don't disregard the possibility that the assumption of the distribution being maintained throughout the file can hurt the model performance as well.

3.3.1 Results

...

3.4 Copy pointer repositioning strategy

Different strategies can be followed when choosing a copy pointer from which to start copying. We developed 3 approaches, 2 of which being fairly simplistic:

- Oldest pointer first: the chosen copy pointer
- Newest pointer first:
- Most common prediction:

3.4.1 Results

3.5 Copy pointer reposition threshold

. . .

3.5.1 Results

3.6 Verbose output

As a sidenote, the model offers the possibility of outputting the probability distributions and predicitons at each step, or of presenting the overall progress of processing the input file.

For the modes outputting the probabilities, the output can be in human-readable format, presenting whether the reported probabilites were from a guess or a prediction hit/miss in colored output. Alternatively, the output can be presented as comma-separated values with less extraneous characters in machine-readable format, which is useful for data analysis afterwards.

4 Text generator

5 Conclusion

6 References

References