DEPARTAMENTO DE ELECTRÓNICA, TELECOMUNICAÇÕES E INFORMÁTICA

Algorithmic Information Theory (2024/2025)

Lab Work #1

Daniel Pedrinho Nº107378

Afonso Baixo Nº108237

Henrique Coelho Nº108342

11th of March 2025

Abstract

This report explores Finite-Context Models (FCMs) for symbol prediction and text generation, implemented in C++. We analyzed how Average Information Content (AIC) evolves across multiple generations of text and studied the effects of context order (k) and smoothing factor (α) on model performance. Results show AIC generally decreases across generations, indicating simplification of complexity in generated sequences. Parameter testing revealed that lower α values typically yield better model fit, though this relationship depends on the sequence structure and symbol diversity. Our work demonstrates practical applications of information theory to statistical modeling and provides insights for optimizing predictive text models.

1 Introduction

Predicting the next symbol in a sequence stands as a fundamental challenge across data compression, statistical modeling, and predictive analytics. A widely used approach to this problem is the *Finite-Context Model (FCM)*, a type of *Markov model* that estimates symbol probabilities based on past observations. These models play a key role in *text compression, natural language processing, and various predictive tasks* by capturing statistical dependencies within a given dataset.

Data compression extends beyond efficient storage and transmission—it fundamentally relies on developing robust *information models* that accurately characterize the statistical properties of a source. Among various implementation choices, **C++** was selected for this project due to its *high performance, low-level memory control, and efficient Standard Template Library (STL)*, which facilitates the manipulation of complex data structures essential for modeling statistical dependencies.

This project focuses on two primary objectives:

1. **Developing a Finite-Context Model (FCM)** to measure the information content of texts, enabling the estimation of *average information per symbol* based on statistical dependencies.

2. **Creating an automatic text generator**, which learns a model from input sequences and generates text that statistically mimics the given source.

A key aspect of this study is analyzing how different parameters—such as *model order* (k) and *smoothing factor* (α)—influence predictive accuracy and compression efficiency. By varying

these parameters, we aim to understand how effectively our model captures *statistical patterns* in different text sources and how compression methods can be optimized.
Ultimately, this exploration will provide deeper insights into *algorithmic information modeling* and *prediction techniques*, demonstrating how different model configurations impact the ability to reproduce characteristic patterns in textual data.

2 Theoretical Background

2.1 Information Models for Prediction

Effective information models capture the statistical regularities within symbol sequences, allowing us to measure the predictability of each symbol based on its context. While various modeling approaches exist, probabilistic models are particularly valuable as they can estimate symbol likelihoods, adapt to different context lengths, and handle unseen sequences through smoothing techniques. These capabilities enable both practical applications in data compression and deeper theoretical insights into the structure of information sources.

2.2 Markov Models and Finite-Context Models

A Markov model is a statistical model that assumes the probability of a future event depends only on a finite number of past events. More formally, a first-order Markov model satisfies the property:

$$P(x_n|x_{n-1}, x_{n-2}, \ldots, x_1) = P(x_n|x_{n-1}). \tag{1}$$

This equation states that the probability of the next symbol x_n depends only on the immediately preceding symbol x_{n-1}, rather than the entire sequence history.
Finite-context models extend this idea by considering a fixed-length context of size k. A k-order finite-context model predicts the next symbol x_n based on the preceding k symbols. This can be mathematically expressed as:

$$P(x_n|x_{n-1}, x_{n-2}, \ldots, x_{n-k}) = \frac{N(x_n|c)}{\sum_{s\in\Sigma} N(s|c)}, \tag{2}$$

where $c = (x_{n-1}, x_{n-2}, \ldots, x_{n-k})$ represents the context and $N(x_n|c)$ denotes the number of occurrences of symbol x_n given the context c. The denominator ensures that the probability distribution is normalized across all possible symbols in the alphabet Σ.

2.3 Probability Estimation and Smoothing Techniques

In practical implementations, probability estimation is performed by counting occurrences of symbols in given contexts. However, a challenge arises when a context has never been observed in the training data, leading to zero probability estimates. This is known as the zero-frequency problem.

To mitigate this issue, smoothing techniques are applied. A common approach is Laplace smoothing, which introduces a small additive constant α to each count:

$$P(e|c) \approx \frac{N(e|c) + \alpha}{\sum_{s \in \Sigma} N(s|c) + \alpha|\Sigma|}. \tag{3}$$

By incorporating the smoothing parameter α, the model ensures that no probability estimate is ever zero, thereby improving generalization to unseen sequences.

2.4 Average Information Content and Entropy

The efficiency of a finite-context model can be evaluated using the concept of average information content. Given a sequence of length n, the average information content of the sequence as estimated by the model is given by:

$$H_n = -\frac{1}{n} \sum_{i=1}^{n} \log_2 P(x_i|c), \tag{4}$$

where $P(x_i|c)$ represents the estimated probability of symbol x_i occurring given its context c. The average information content is measured in bits per symbol (bps), and a lower value indicates a more efficient model, as it implies a better prediction of the sequence.

Entropy is a measure of the uncertainty or randomness in a system and is calculated as:

$$H = -\sum_{i} p_i \log_2 p_i \tag{5}$$

where p_i represents the probability of a symbol i occurring in the sequence. Entropy quantifies the expected information content or uncertainty across the entire probability distribution.

2.5 Applications of Finite-Context Models

Finite-context models have been applied in a wide range of fields due to their ability to capture local dependencies in sequential data. Some key applications include:

- **Data Compression:** Finite-context models are widely used in text and data compression algorithms. Techniques such as Prediction by Partial Matching (PPM), Lempel-Ziv-Prediction (LZP), and the Burrows-Wheeler Transform (BWT) rely on finite-context modeling principles to achieve efficient compression [1].
- **DNA Sequence Analysis:** These models have been successfully applied to DNA sequence analysis. They are used for compressing DNA data and generating synthetic DNA sequences, which are valuable for genomic studies [2].

These applications demonstrate the versatility of finite-context models in handling structured data across diverse fields.

3 Methodologies

3.1 Finite-Context Model

The Finite-Context Model (FCM) is designed to estimate the probability of symbols occurring in a given sequence based on past observations. It relies on a Markovian assumption where the probability of a symbol appearing is conditioned on a fixed-length context window of prior symbols. This approach allows capturing statistical dependencies within the data while maintaining computational efficiency.

To construct the model, the text is parsed to extract all possible contexts of length k. For each extracted context, the model keeps track of the number of occurrences of each subsequent symbol. Additionally, a separate context count table is maintained to facilitate probability estimation.

Given a context c, the probability of a symbol s appearing is computed using the following formula:

$$P(e|c) \approx \frac{N(e|c) + \alpha}{\sum_{s \in \Sigma} N(s|c) + \alpha|\Sigma|}. \tag{6}$$

where:

- *N(e/c)* represents the number of times symbol e appears after context c.
- $\sum_{s \in \Sigma}$ *N(s/c)* is the total amount of symbols given the context c.
- α is the smoothing parameter to handle unseen symbols.
- $|\Sigma|$ represents the total number of unique symbols in the dataset.

To ensure numerical stability and account for data sparsity, a smoothing parameter α is introduced. Without smoothing, unseen contexts would have zero probability, leading to undefined results during entropy calculations.

The effectiveness of the FCM is evaluated by computing the average information content of the text, and is given by:

$$H_n = -\frac{1}{n} \sum_{i=1}^{n} \log_2 P(x_i|c), \tag{7}$$

where *P(x_i/c)* represents the estimated probability of symbol x_i occurring given its context c. A lower AIC value suggests a better model fit, meaning the model accurately captures the statistical patterns of the text.

Once trained, the model is saved to a text file containing the context-symbol frequency table, the order k, and the smoothing parameter α. This allows us to store the gathered knowledge and save it for later use.

3.2 Generator

The generator takes as input the frequency table generated previously. This table comes in the format of a .txt file, and also contains the order of the model and its smoothing factor, according to the parameters used in the FCM. The table is read as an unordered map.

The text generation is based on the weights of each character according to the current context. The context is read and, according to it, a weighted random selection is made to generate the next character. This character is then appended to the initial context, the "prior", given as an input parameter.

Then, the context window moves one character over, to include the newly generated character, and repeats the generation process. This process is repeated until the given generation size is reached. Once the given size is reached, the generated text is printed out to a .txt file.

4 Experimental Results and Analysis

In this section, we analyze the experimental results obtained from the two different tests conducted on the sequences. The first test examines the growth of Average Information Content (AIC) values, in bps (Bits per Symbol), across successive generations, while the second test evaluates AIC under different parameter settings for α and order k.

4.1 AIC Evolution Across Generations

To investigate the evolution of AIC across generations, we generated successive sequences by iteratively producing new sequences from their predecessors. This process continued for up to 10 generations. The results for sequence 1 are illustrated in Figure 1, demonstrating how AIC changes as generations progress.

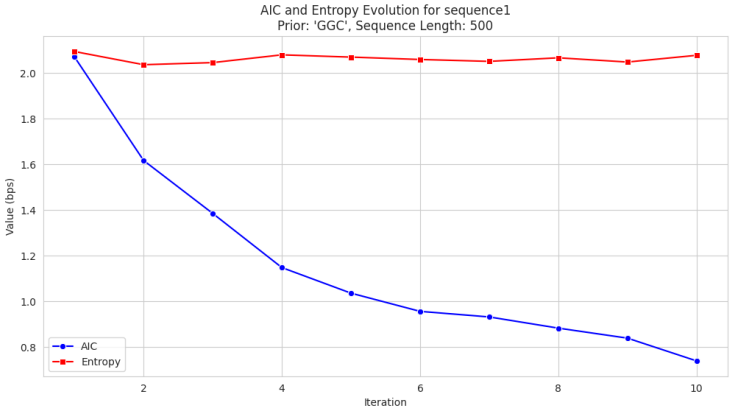

Figure 1: AIC evolution over generations for sequence 5 ($k = 3$ & $\alpha = 0.1$).

Similar trends were observed across the other sequences, as shown in Figures of sequences 2 to 5

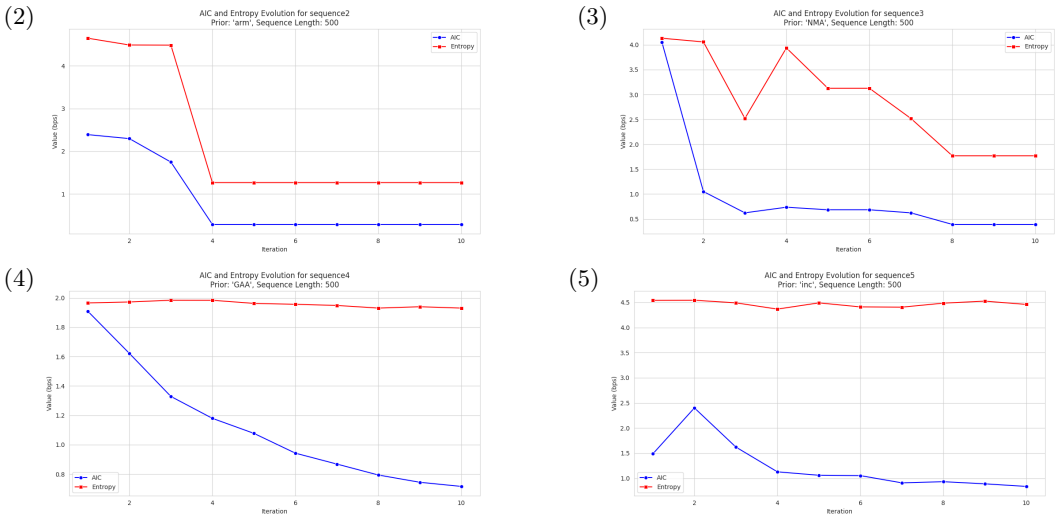

Figure 2: AIC evolution over generations for sequences 2 to 5 ($k = 3$ & $\alpha = 0.1$).

As observed, AIC values generally exhibit a decreasing trend over generations, suggesting that as new generations are born, their complexity becomes simpler. The decreasing pattern varies across different sequences, indicating possible dependencies on initial conditions and structural properties of the sequences.

4.2 AIC for Different Parameter Settings

In the second experiment, we analyzed the effect of different values of α and order k on the AIC values. This test provides insights into how parameter selection influences the model's performance and generalization capability. The results for sequence 1 are presented in Figure 3.

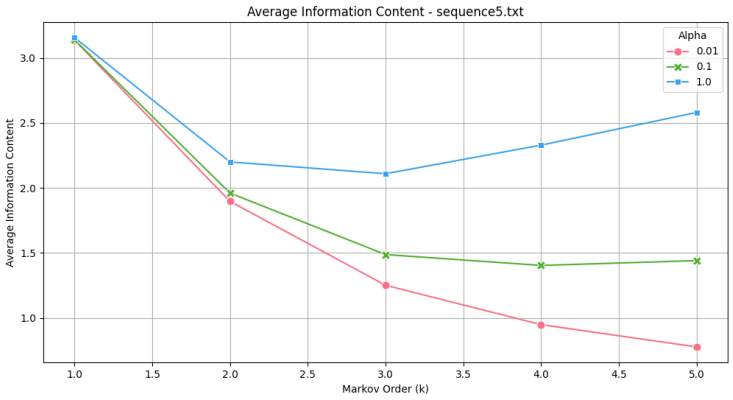

Figure 3: AIC values for different α and order k configurations for sequence 1.

Similar tests were conducted for sequences 2 to 5, with results shown in Figure 4.

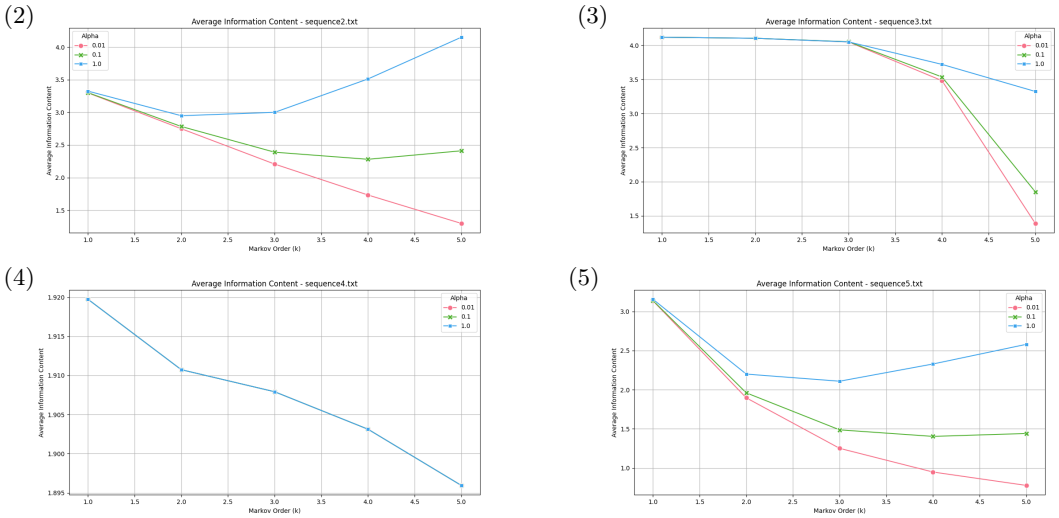

Figure 4: AIC values for different α and order k configurations for sequences 2 to 5.

From these results, we observe that AIC values fluctuate significantly depending on the choice of parameters. We can visually grasp that as k value increases, so does the fluctuation between values of α. Lower values of α lead to lower AIC scores, indicating better model fit, whereas for other sequences. This suggests that the optimal selection of parameters, in order to obtain a lower AIC score, requires lower values of α, yet, as we can see in sequence4, the AIC does not change whatsoever for different values of alpha. This tells us that this fluctuation also depends on the structure of the sequence it self and on the amount of unique symbols $|\Sigma|$ it contains.

5 Conclusion

This study demonstrates that Finite-Context Models effectively captures statistical patterns in text data. Our experiments revealed that AIC decreases across generations, indicating generated sequences become progressively less complex through iterative generation. This simplification likely results from reinforcement of dominant patterns while unique features become diluted.

Our parameter analysis showed that lower α values generally produced better model fits, but optimal configurations varied significantly across different sequences, highlighting the importance of tuning parameters based on source material characteristics.

Entropy measurements supported our AIC findings, showing a similar but less pronounced decreasing trend across generations. This suggests that while sequences become more predictable, the underlying symbol distribution remains relatively stable.

Overall, these findings provide valuable insights into AIC behavior under different evolutionary processes and parameter configurations, which can be directly applied to optimize sequence modeling strategies for applications in data compression and text generation. Future work could explore adaptive parameter selection and methods to preserve complexity across multiple generations.

References

[1] D. Phong, "Finite context modelling", *Hugi*, no. 19, n.d. Retrieved from `https://hugi.scene.org/online/coding/hugi%2019%20-%20cofinite.htm`.

[2] D. Pratas, A. J. Pinho, A. J. R. Neves, and C. A. C. Bastos, "Dna synthetic sequences generation by finite-context models", *Signal Processing Lab, DETI / IEETA University of Aveiro, Portugal*, 2010, Available: `https://sweet.ua.pt/pratas/papers/Pratas-2010a.pdf`.