Finite-Context Models for Text Prediction and Generation: Experimental Analysis

André Oliveira
University of Aveiro
107637
Email: andreaoliveira@ua.pt

Francisco Ferreira
University of Aveiro
124467
Email: ftferreira@ua.pt

Alexandre Cotorobai
University of Aveiro
107849
Email: alexandrecotorobai@ua.pt

***Abstract*—This report presents an experimental analysis of finite-context models (FCMs) for text prediction and generation. The focus is on evaluating model performance across varying parameters, such as model order k and smoothing parameter α, using several text sequences. Our results, which include aggregated performance tables, heatmaps, and facet grids, demonstrate the trade-offs between model complexity and predictive accuracy.**

***Index Terms*—Finite-Context Models, Markov Models, Data Compression, Text Generation, Algorithmic Information Theory.**

1. Introduction

Text prediction and generation are critical problems in data compression and natural language processing. Finite-context models (FCMs) are used to capture high-order dependencies in textual data by estimating symbol probabilities based on preceding contexts. This paper describes the development of two programs: `fcm` for computing the average information content of texts and `generator` for generating text based on learned models. We also discuss parameter tuning and present experimental results for various sequences.

2. Background and Related Work

The use of Markov models for text prediction has been explored extensively since Shannon's seminal work. However, the direct application of high-order models is hindered by the exponential growth in model size. In this work, we employ smoothing techniques to overcome the zero-frequency problem in probability estimation.

3. Methodology

In our implementation, the order of the model k and the smoothing parameter α are adjustable via command-line parameters. The `fcm` program computes the average information content in bits per symbol (bps) using the formula:

$$H_n = -\frac{1}{n}\sum_{i=1}^{n}\log_2 P(x_i \mid c)$$

where c is the context of k preceding symbols.

The `generator` program generates text using the precomputed Finite Context Model (FCM).It takes a trained model as input and probabilistically predicts the next symbols based on the observed context.

Given a prior sequence of length k, the program iteratively selects the next character using frequency distributions stored in the model. The probabilities are smoothed using the parameter α to handle unseen contexts. The generated sequence can then be analyzed to assess compression efficiency.

3.1. Tuning Parameters Approaches

In our experiments regarding the tuning of parameters k and α, our initial objective was to minimize the bits per symbol (bps), aiming for optimal compression efficiency. However, through careful observation of the generated text and subsequent analysis presented later, we realized that solely focusing on minimizing bps might lead to unintended losses in structured information.

When applying these findings specifically to the text from `sequence2.txt` we explored the possibility of using the average word length as context. Although this seemed promising for textual data, we quickly recognized that this approach would not translate effectively to DNA sequences due to their fundamentally different structural properties.

Consequently, we considered alternating between two models: one utilizing a higher-order context and another of order-1. Unfortunately, this approach encountered practical difficulties, as the introduction of the order-1 model often disrupted the ability to find corresponding context entries in the higher-order model. This led to a problematic loop, indicating the necessity for additional mechanisms to facilitate efficient model switching, which will be discussed in greater depth later.

To solve this problem, we incorporated Levenshtein distance, a metric that measures the minimum number of single-character edits (insertions, deletions, or substitutions) required to transform one string into another. By applying this technique, our generator was able to identify and substitute the closest available context when an exact match was unavailable. This allowed the system to generate previously

unseen words by finding the most similar existing words, thereby enhancing diversity while preserving structural coherence.

Lastly, we contemplated comparing the bps of models generated from the original text with those from models trained on previously generated text, an idea aligned with the study elaborated subsequently. This comparative approach became the primary focus and driving concept for our current work.

4. BPS Comparision Studies

Bits per symbol (BPS) in finite-context (Markov) models represent the average cross-entropy or log-loss per symbol. If a generative model perfectly captures the original text distribution, its BPS will approach the actual source entropy. Significant differences in BPS between original and generated text indicate divergences between the model's learned distribution and the true distribution of the text.

4.1. Low BPS Variance → Good Model Fit

A model with small variance in BPS between original and generated texts accurately replicates the statistical properties of the original text. From an information-theoretic perspective, the true text distribution minimizes cross-entropy over the data. Therefore, a model whose BPS closely matches the original text's entropy effectively captures data predictability.

The average code length (in bits per symbol) is minimized when the model's distribution matches the source distribution exactly. Any deviation introduces redundancy (additional bits per symbol). Thus, a small BPS difference between original and generated text implies minimal redundancy, indicating good model fit.

Perplexity, directly related to BPS, is a standard metric for evaluating language models. Classic experiments by Shannon demonstrated that increasing context order (higher-order Markov models) significantly reduces BPS. For example, contexts of up to 8 letters yield about 2.3 bits/letter, whereas very long contexts ($\approx$ 100 characters) reduce entropy to around 1.0 bit/letter. A model achieving around 1 bit/letter is closer to the true distribution than models stuck around 2–4 bits/letter.

Modern compression-based models (e.g., PPM) illustrate that better modeling (longer contexts and smarter strategies) leads to lower BPS. Teahan and Cleary (1996) [3] showed a PPM model compressing text to 1.46 bits/character, significantly improving over trigram models ($\approx$1.75 bits/character), reflecting a more faithful capture of English statistics.

However, it is important to note that a small BPS difference only guarantees fidelity within the model's context scope. A model may accurately capture short-range statistics (yielding low BPS) but fail at long-range coherence. For instance, a first-order Markov model perfectly matching single-letter frequencies will have similar entropy to the original text at the single-letter level but will produce incoherent output.

In general, for sufficiently comprehensive Markov models, a low BPS variance reliably indicates the model accurately mirrors the original text distributions.

4.2. High BPS Variance → Missing Source Structure

A large variance or gap in bits-per-symbol (BPS) between original and generated text indicates that a model has failed to capture essential aspects of the source, leading to discrepancies in predictive performance. Practically, this manifests as output that's either too random or too repetitive compared to the original, signaling a mismatch detected by BPS analysis.

When Markov models neglect certain dependencies present in the source text, they inaccurately treat symbols as more unpredictable. This raises cross-entropy on the original text, indicating surprise at patterns that actually exist, and results in less efficient compression. Shannon's [1] experiments showed that finite-context models missing long-range structure encode text at higher bits-per-character (e.g., $\approx$ 2.3 bits/char) than models capturing full structure ($\approx$ 1.0 bit/char), demonstrating significant unmodeled redundancy (e.g., syntax, semantics). High BPS relative to the source thus signals underfitting, with critical patterns absent and predictive errors higher.

Conversely, models might also over-generalize, creating nonsensical or overly repetitive sequences. Evaluating these sequences under true distributions or robust models yields extreme BPS values, reflecting inconsistencies between model outputs and the source distribution. Such differences are often identified using compression-based classification techniques, highlighting model mismatches through abnormal code lengths.

Quantitatively, large BPS differences represent significant KL divergence, penalizing model-source distribution mismatches. High variance in BPS indicates neglect of long-range contexts, rare events, or incorrect frequency assignments, commonly due to short contexts or poor smoothing. Research by Thaper et al. (2001) [4] noted that compression metrics can distinguish original from machine-generated texts due to violations of long-range patterns by finite-context models.

In summary, significant variance or gaps in BPS clearly indicate that a model fails to replicate essential structural and statistical regularities of the source text, harming predictive accuracy and compressibility.

5. Experimental Results

Experimental evaluation was performed on a set of sequences (sequence1.txt, sequence2.txt, ..., sequence5.txt). For each sequence, the best parameters were determined by tuning k and α. The following subsections detail the results:

5.1. Time Analysis

5.1.1. Aggregated Performance Tables.

Tables 1 to 5 summarize the performance metrics (Mean Time, Std Dev, Min, and Max Time) averaged over different values of α for each k per execution of our pipeline. The pipeline involved computing the BPS of the input sequence using FCM, generating text based on the calculated model, and then recalculating the BPS for the newly generated sequence.

TABLE 1. AGGREGATED PERFORMANCE METRICS FOR SEQUENCE1.TXT

k	Mean Time (s)	Std Dev (s)	Min Time (s)	Max Time (s)
3	0.0427	0.0224	0.0258	0.0660
5	0.0355	0.0047	0.0320	0.0407
10	0.0642	0.0111	0.0543	0.0758
15	0.0598	0.0082	0.0525	0.0679
20	0.0634	0.0049	0.0591	0.0685

TABLE 2. AGGREGATED PERFORMANCE METRICS FOR SEQUENCE2.TXT

k	Mean Time (s)	Std Dev (s)	Min Time (s)	Max Time (s)
3	0.3236	0.0111	0.3142	0.3352
5	0.6231	0.0155	0.6072	0.6380
10	1.1696	0.0092	1.1593	1.1768
15	1.2350	0.0151	1.2199	1.2482
20	1.3168	0.0148	1.3033	1.3311

TABLE 3. AGGREGATED PERFORMANCE METRICS FOR SEQUENCE3.TXT

k	Mean Time (s)	Std Dev (s)	Min Time (s)	Max Time (s)
3	4.9601	0.1181	4.8622	5.0895
5	10.3181	0.2705	10.1415	10.6175
10	14.3789	0.6849	13.6056	14.8799
15	13.8288	0.0842	13.7479	13.9121
20	14.5411	0.0768	14.4735	14.6227

TABLE 4. AGGREGATED PERFORMANCE METRICS FOR SEQUENCE4.TXT

k	Mean Time (s)	Std Dev (s)	Min Time (s)	Max Time (s)
3	12.4171	0.1201	12.2866	12.5158
5	13.0574	0.1548	12.9389	13.2305
10	39.8358	0.6115	39.3745	40.4868
15	80.9396	0.8183	80.0895	81.6995
20	90.0095	2.3346	88.3239	92.6154

5.1.2. Heatmap of Mean Time.

For better comprehension, we also analyzed the time performance using heatmaps. The heatmap for Sequence2 is shown in Figure 1. The vertical axis represents the smoothing parameter α, while the horizontal axis represents the model order k. Each cell displays the mean execution time in seconds.

TABLE 5. AGGREGATED PERFORMANCE METRICS FOR SEQUENCE5.TXT

k	Mean Time (s)	Std Dev (s)	Min Time (s)	Max Time (s)
3	0.4911	0.0077	0.4852	0.4991
5	0.4800	0.0079	0.4721	0.4874
10	0.6976	0.0126	0.6869	0.7109
15	0.9099	0.0100	0.8997	0.9190
20	1.0826	0.0139	1.0705	1.0967

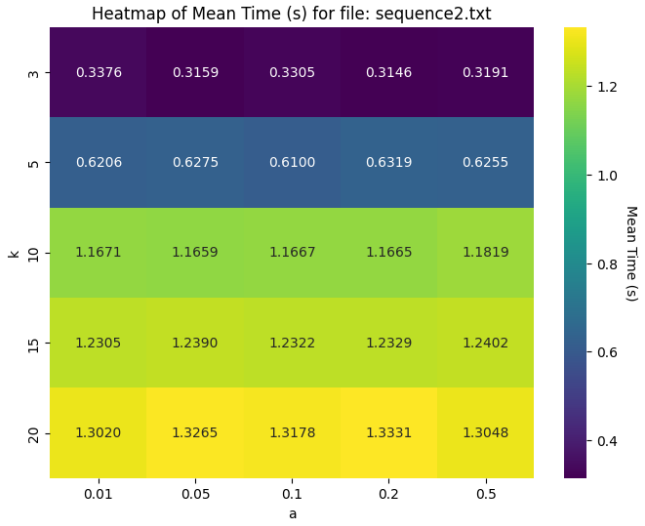

Figure 1. Heatmap of Mean Time (s) for Sequence2.txt.

As observed, increasing k results in higher execution times for our pipeline, while variations in α have little impact on performance. This trend is consistent across all sequences.

5.2. BPS Comparison Experiments

In our experiments, we executed the finite-context model (FCM) twice:

1) **Original Text Analysis:** First, we ran the FCM on the original text to compute the average Bits Per Symbol (BPS). This value serves as a baseline that quantifies the model's ability to predict the source text.
2) **Generated Text Analysis:** Next, we used the model, trained on the original text, to generate a new text sequence. We then re-ran the FCM on this generated text to compute its average BPS.

In this section, we analyze the experimental results obtained from applying finite-context models (FCMs) on two different textual sequences: one representative of DNA data (sequence 1) and another derived from classical Portuguese literary text - "Os Lusíadas" (sequence 2). Specifically, we assess the impact of model order (k) and smoothing factor (α) on bits per symbol (BPS).

5.2.1. Sequence 1: DNA Data.

Applying FCMs to DNA sequences demonstrates a clear decrease in BPS values for both original and generated texts as the context order (k) increases. At higher orders ($k \geq 10$), BPS values approach zero, signifying highly accurate model predictions and near-perfect replication of original sequence statistics as seen in Figure 2. Additionally, the smoothing factor (α) has limited influence on DNA sequence performance, with negligible BPS differences at higher k and α values.

Heatmap visualization (Figure 3) further corroborates these findings, displaying minimal BPS differences ($|bps_1 - bps_2| \approx 0$) at context lengths of $k \geq 15$, thus highlighting the model's robust predictive capabilities.

5.2.2. Sequence 2: Portuguese Literary Text.

Contrastingly, the Portuguese literary text displays significantly different sensitivity to model parameters. Here, substantial BPS variations are observed, especially at lower smoothing factors and shorter contexts ($k < 5$). Optimal results, characterized by moderate BPS differences (≈ 0.107), are attained at intermediate parameter settings ($k = 5$, $\alpha = 0.1$).

Interestingly, increasing context lengths beyond certain thresholds ($k \geq 15$) combined with higher smoothing parameters (α) paradoxically leads to increased BPS variance (Figure 4), suggesting diminished predictive efficacy due to excessive generalization. Thus, optimal predictive accuracy in literary texts necessitates moderate context lengths ($5 \leq k < 10$) paired with intermediate smoothing values ($0.05 \leq \alpha \leq 0.1$).

Heatmap analysis (Figure 5) distinctly highlights these optimal parameter zones, illustrating reduced BPS differences within clearly defined areas. These insights emphasize meticulous parameter selection for textual data exhibiting complex linguistic and semantic characteristics.

5.2.3. Summary of Findings.

The comparative analysis underscores differing parameter tuning requirements based on textual nature. DNA-like sequences, characterized by structured, repetitive data, benefit from high-order contexts achieving minimal BPS differences. In contrast, natural language texts with intricate structures require moderate contexts and balanced smoothing to achieve optimal predictive accuracy without overfitting.

5.3. Complexity Profiling

In addition to the BPS comparisons, we used a complexity profiler to analyze the evolution of BPS along the text sequence. This profiler provides a dynamic view of the model's performance by tracking the bits per symbol across different segments of the text. Key aspects include:

- **Local Variations:** The profiler reveals specific regions where the model's predictive performance fluctuates, highlighting areas of higher or lower complexity.
- **Trend Analysis:** The overall trend of BPS across the sequence indicates whether the model consistently fits the text or deviates in certain segments.

5.4. Validation and Complexity Profile Analysis

To validate the accuracy of our complexity profile implementation, we first tested the program using a 50-character sequence provided in class (Figure 6). Our objective was to ensure that our results closely matched the expected reference complexity profile. After executing our program on this sample sequence, we verified that the generated output matched exactly with the provided reference, confirming the correctness and reliability of our implementation.

Following the validation phase, we conducted a detailed analysis of complexity profiles by varying the model order parameter k. This allowed us to examine how different context lengths influenced entropy and predictability within the generated profiles.

5.5. Effect of Different k Values on Complexity Profile

5.5.1. $k = 1$ (Low-Order Context) (Figure 7).

- The complexity profile shows high variance, characterized by significant fluctuations in bits-per-symbol throughout the sequence.
- This behavior indicates difficulty in accurate symbol prediction when using a first-order Markov model, primarily due to insufficient contextual information.
- Consequently, entropy remains elevated in the initial portions of the sequence.

5.5.2. $k = 3$ (Moderate-Order Context) (Figure 8).

- The complexity profile begins to stabilize in certain regions, although some fluctuations persist, particularly at the sequence's beginning.
- Initial elevated entropy values reflect uncertain predictions. However, as the context length increases, the bits-per-symbol metric progressively smoothens.
- This illustrates that a moderate increase in context length improves predictive accuracy, although unpredictability persists in certain regions.

5.5.3. $k = 5$ (Higher-Order Context) (Figure 9).

- The complexity profile becomes notably smoother, exhibiting a substantial reduction in entropy following an initial fluctuating region.

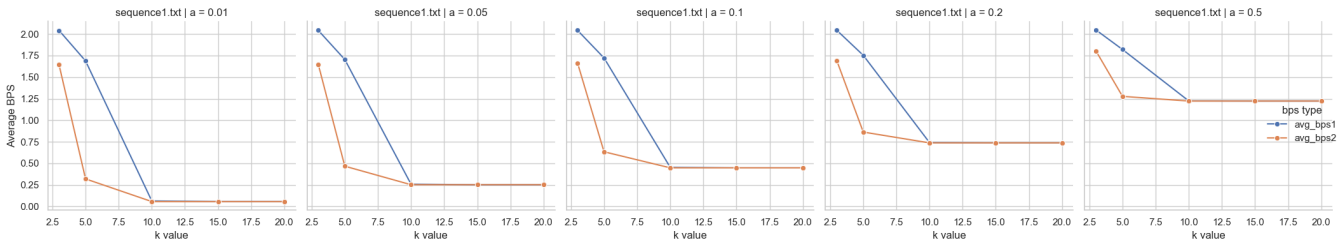

Figure 2. BPS comparison graph between default text and generated text

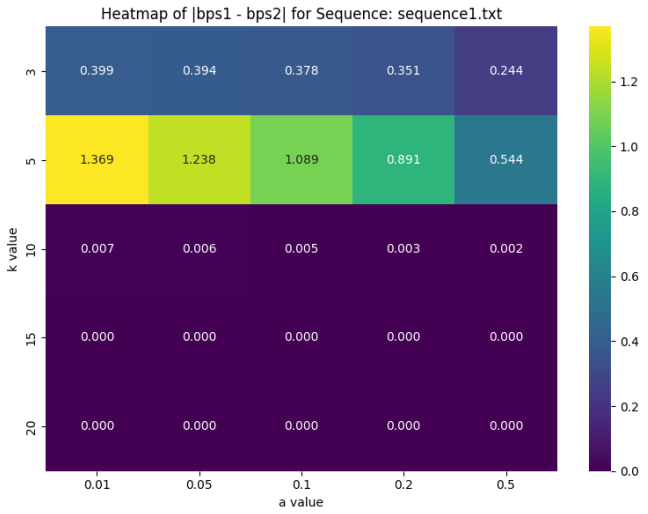

Figure 3. BPS comparison heatmap

- The latter portion of the sequence achieves near-zero entropy values, indicating high confidence in model predictions.
- These results clearly demonstrate that higher-order contexts effectively capture sequence structures, thereby substantially reducing uncertainty.

Overall, our analysis successfully validates the effectiveness of our complexity profiling approach, highlighting the critical role that context size plays in influencing entropy and predictive accuracy across diverse sequence types.

5.6. Complexity Profile Comparison: Original vs. Generated Sequences

To further evaluate the predictive capabilities and fidelity of our finite-context model (FCM), we compared the complexity profiles of an original textual sequence (Sequence 5), which consists of C programming language source code, against a sequence generated by our FCM implementation. Complexity profiles were analyzed using context lengths (k) of 1 and 5, with a smoothing parameter ($\alpha = 0.01$) previously determined to be optimal.

5.6.1. Low-Order Context ($k = 1$).

At a context length of $k = 1$, both original (Figure 10) and generated (Figure 11) sequences showed relatively high entropy values (bits per symbol), accompanied by considerable fluctuations and variance across their complexity profiles. The observed differences between these two profiles highlight the limitations of low-order contexts, indicating the model's difficulty in capturing meaningful patterns from minimal contextual information. Consequently, the generated sequence diverged significantly from the structural properties inherent in the original source code.

5.6.2. Optimal Higher-Order Context ($k = 5$).

When increasing the model order to $k = 5$ (Figure 12) and (Figure 13), significant improvements emerged:

- Both complexity profiles exhibited substantially lower and more consistent entropy values, reflecting enhanced predictive accuracy.
- The generated sequence's complexity profile closely aligned with that of the original sequence, demonstrating effective capture and reproduction of the original code's statistical and structural properties.
- This alignment emphasizes that a moderately high context length enables the model to learn longer-range dependencies within source-code data, reducing predictive uncertainty.

5.6.3. Insights and Implications.

The comparative analysis underscores the importance of selecting appropriate parameters when using finite-context models for structured textual data, such as programming languages. Low-order models ($k = 1$) were insufficient for accurately modeling such structured data. Conversely, optimal parameter settings ($k = 5$ with $\alpha = 0.01$) allowed the model to successfully capture deeper contextual dependencies, significantly reducing entropy and resulting in generated sequences whose complexity closely resembled the original text.

5.7. Entropy Interpretation and Predictability

The Shannon entropy $H(X)$ of sequence 4 was calculated as 1.9652, which remains constant across different values of k. This indicates that the overall randomness of

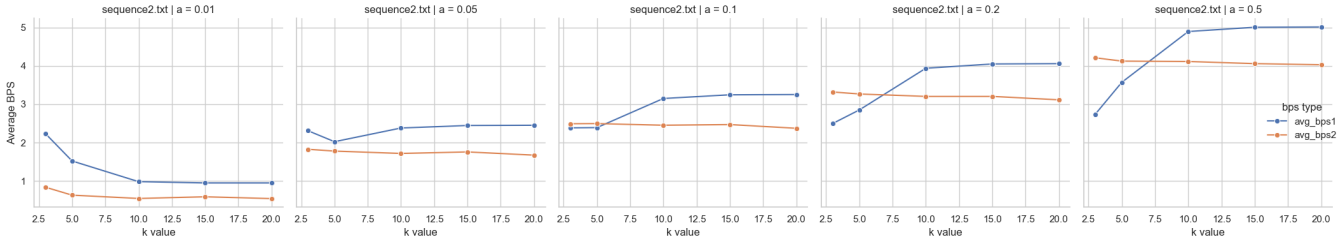

Figure 4. BPS comparison graph between default text and generated text

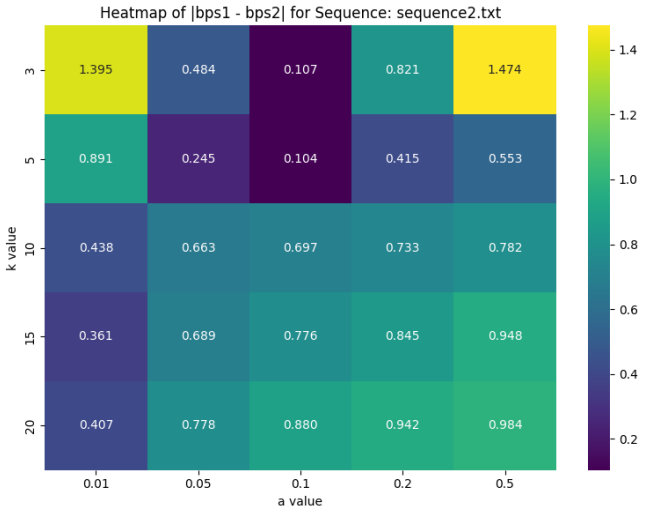

Figure 5. BPS comparison heatmap

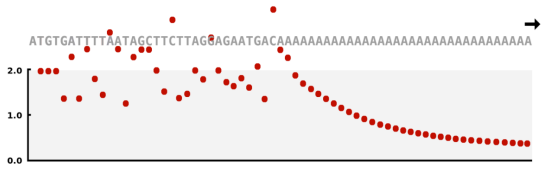

Figure 6. Complexity Profile Class Example

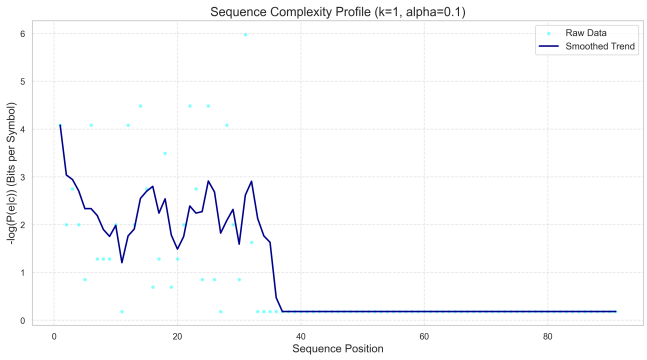

Figure 7. Complexity Profile generated for class example, k=1

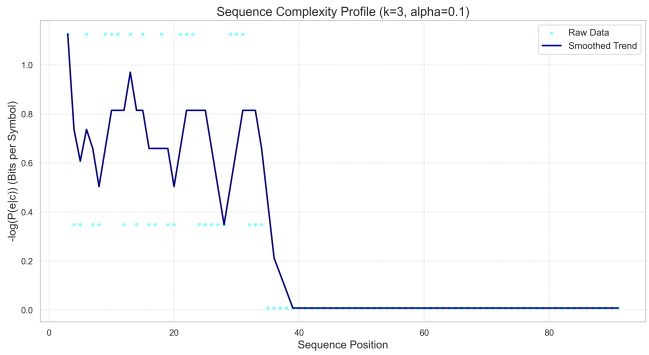

Figure 8. Complexity Profile generated for class example, k=3.

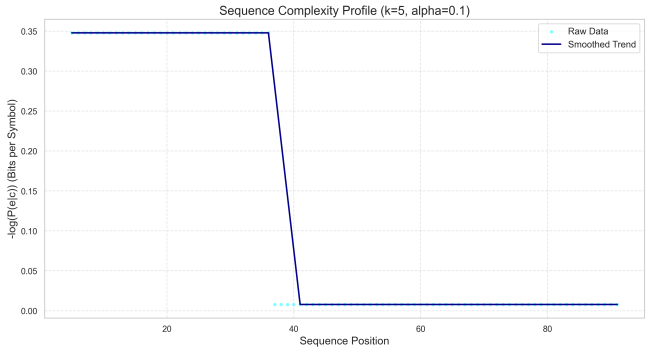

Figure 9. Complexity Profile generated for class example, k=5.

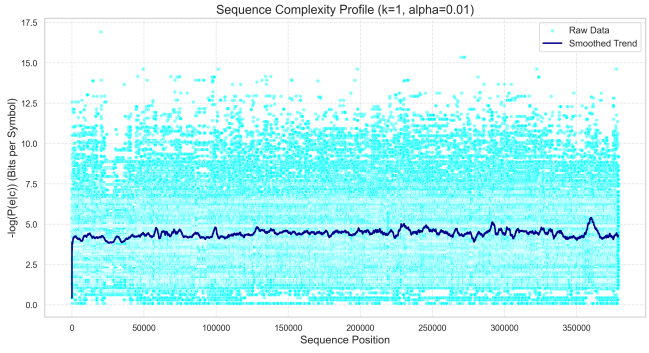

Figure 10. Complexity Profile of sequence 5, k=1

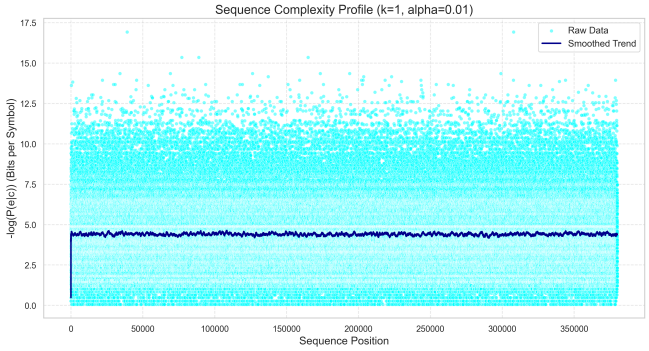

Figure 11. Complexity Profile of generated text from sequence 5, k=1

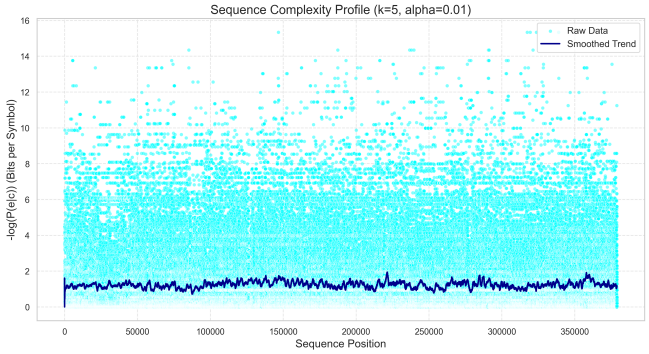

Figure 12. Complexity Profile of sequence 5, k=5

the sequence is inherent and does not change regardless of the context length considered.

Shannon entropy is given by:

$$H(X) = -\sum_{x \in X} p(x) \log_2 p(x) \tag{1}$$

where $p(x)$ is the probability of occurrence of each symbol x in the sequence.

However, the conditional entropy $H(Y|X)$, which measures the uncertainty of predicting the next symbol given a context of length k, exhibits a significant drop as k increases:

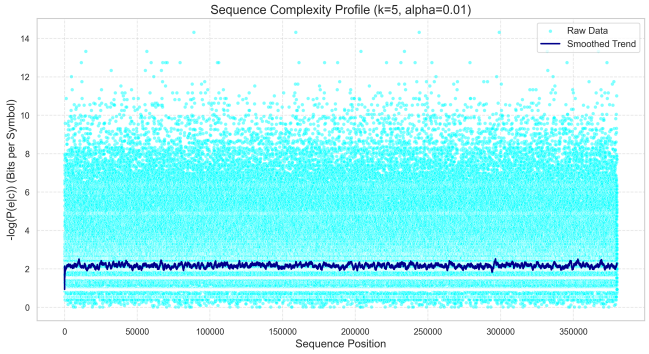

Figure 13. Complexity Profile of generated text from sequence 5, k=5

$$H(Y|X) = -\sum_{x \in X} p(x) \sum_{y \in Y} p(y|x) \log_2 p(y|x) \tag{2}$$

where $p(y|x)$ represents the probability of a symbol y occurring given the previous context x.

- At $k = 1$, $H(Y|X) = 1.9198$, meaning there is still a high level of uncertainty when predicting the next symbol using only a single preceding symbol.
- At $k = 15$, $H(Y|X) = 0.1356$, demonstrating that with longer contexts, the sequence becomes much more predictable.

5.7.1. Redundancy (R) and Predictability.

The redundancy R, which quantifies how much of the sequence is predictable, is computed as:

$$R = 1 - \frac{H(Y|X)}{H(X)} \tag{3}$$

It shows a stark contrast between small and large context lengths:

- $R = 0.0231$ at $k = 1$: Very low redundancy, meaning the sequence appears highly random when considering only one previous symbol.
- $R = 0.9310$ at $k = 15$: High redundancy, suggesting that most of the sequence's structure is captured at this context length, making it highly predictable.

5.7.2. Relation to BPS Analysis and Optimal Context Length.

The BPS values computed for sequence 4 further validate our findings. At $k = 1$, we obtained a BPS value of 1.91975, which closely matches the conditional entropy $H(Y|X) = 1.9198$. This indicates that at lower context lengths, the sequence remains highly unpredictable. However, as we increase the context length to $k = 15$, the BPS value drops significantly to 0.380467, aligning with the sharp reduction in conditional entropy to $H(Y|X) = 0.1356$.

This strong correlation between BPS and conditional entropy reinforces our previous conclusions: higher k values lead to more accurate predictions, reducing uncertainty and increasing redundancy. The results demonstrate that the FCM successfully captures and replicates the statistical properties of the sequence.

Furthermore, our earlier parameter optimization experiments identified $k = 15$ as an optimal context length for this sequence. The fact that both the BPS metric and entropy measures show significant improvements at this value confirms its effectiveness in modeling DNA sequences.

5.8. Comparison with Zstandard Compression

In this section, we compare the performance of our compression pipeline, as represented by the FCM-based method, against the commonly used Zstandard (ZIP) compression. Table 6 presents a summary of the compression statistics for each sequence, including the compressed size in bits, the number of characters, and the bits per symbol (BPS) for both the ZIP and FCM-based compression methods.

The BPS values for ZIP compression (labeled as ZIP BPS) and FCM compression (labeled as FCM BPS) are presented for each sequence. It is evident from the table that, in general, the FCM method provides a more compact representation with lower bits per symbol compared to ZIP compression, especially for Sequence1 and Sequence2. However, for Sequence3 and Sequence4, the difference between the two methods is smaller.

TABLE 6. COMPRESSION AND FCM STATISTICS FOR SEQUENCES

Sequence	Size (bits)	Chars	ZIP BPS	FCM BPS
Sequence1	28,072	10,126	2.7723	2.04688
Sequence2	1,065,056	318,185	3.3473	2.73718
Sequence3	13,837,896	3,295,751	4.1987	4.0508
Sequence4	48,725,896	22,668,225	2.1495	1.90792
Sequence5	651,056	378,930	1.7181	1.85937

Overall, our results suggest that FCM compression can offer competitive performance when compared to Zstandard, particularly for smaller datasets, where it consistently shows lower BPS values. In larger sequences, both methods approach similar performance, but FCM maintains a slight edge in terms of compression efficiency. This comparison highlights the potential of FCM as a promising alternative to traditional compression methods, especially for certain types of textual data.

6. Discussion

The results presented in this study clearly demonstrate that the effectiveness of finite-context models (FCMs) is strongly influenced by parameter selection, specifically the model order (k) and the smoothing factor (α). We observed that as the context k increases, there is a significant reduction in the uncertainty of predictions made by the models, as reflected by the consistent decrease in Bits per Symbol (BPS) values. This phenomenon indicates that higher-order models can capture deeper statistical dependencies, significantly improving their predictive capability.

However, the impact of the analyzed parameters varies substantially depending on the nature of the data. Highly structured and repetitive data, such as DNA sequences, clearly benefit from the use of models with broader contexts ($k \geq 10$), where the BPS reduction is pronounced, and the influence of α is less significant. In contrast, for literary texts, as observed in the analysis of *Os Lusíadas*, excessively high context lengths can compromise predictive performance by inducing excessive generalization. In these cases, moderate contexts (k between 5 and 10) combined with intermediate α values (between 0.05 and 0.1) proved to be more effective.

The complexity profile analysis further confirmed that larger contexts reduce conditional entropy and increase redundancy, significantly improving predictive accuracy. Additionally, the comparison between FCM and ZIP compression methods revealed that finite-context models can offer competitive performance, particularly in smaller datasets with well-defined structure.

7. Conclusion

This study demonstrated the crucial importance of parameter optimization in finite-context models for text prediction and generation. It was observed that the appropriate choice of context length (k) and smoothing factor α is essential for balancing predictive accuracy and computational efficiency. The results emphasize the necessity of carefully tuning models according to the intrinsic structure of the analyzed data, suggesting that adaptive approaches could further maximize the performance of these models.

8. Future Work

Future research will explore the integration of Deep Learning methodologies to facilitate effective alternation between models of varying orders. This approach aims to leverage the advantages of different context-length models dynamically, depending on local textual structures. By using neural network architectures, such as recurrent neural networks (RNNs) or attention-based mechanisms, it may be possible to predict optimal switching points between lower-order and higher-order models, enhancing overall predictive accuracy and coherence in generated text.

Another promising avenue is to develop adaptive smoothing techniques using machine learning algorithms, enabling real-time adjustment of smoothing parameters based on textual complexity and sparsity of context occurrences. Additionally, the creation of binary vectors to pinpoint locations of highly infrequent symbols will be considered, thereby avoiding unnecessary increases in model complexity and ensuring efficient resource utilization.

Furthermore, extending experimental analyses to multilingual datasets and cross-domain text corpora may provide insights into the generalizability of finite-context models and their adaptability to diverse textual characteristics.

Acknowledgments

We would like to thank our teachers for their valuable guidance and the class materials that contributed to the successful completion of this work.

References

[1] C. E. Shannon, "Prediction and Entropy of Printed English," *Bell System Technical Journal*, vol. 30, no. 1, pp. 50–64, 1951.

[2] R. Begleiter, R. El-Yaniv, and G. Yona, "On Prediction Using Variable Order Markov Models," *Journal of Artificial Intelligence Research*, vol. 22, pp. 385–421, 2004.

[3] W. Teahan and J. Cleary, "The entropy of English using PPM-based models," in *Proc. Data Compression Conference (DCC)*, 1996, pp. 53–62.

[4] N. Thaper, "Using Compression for Source Based Classification of Text," M.S. thesis, Massachusetts Institute of Technology, 2001.

[5] "Evaluation Metrics for Language Models," *The Gradient*, 2019. [Online]. Available: https://thegradient.pub/understanding-evaluation-metrics-for-language-models. [Accessed: 12-Mar-2025].

[6] A. J. Pinho, D. Pratas, and S. P. Garcia, "Complexity Profiles of DNA Sequences Using Finite-Context Models," in *Proc. of the International Conference on Bioinformatics and Computational Biology (BIOCOMP)*, 2011.

[7] D. Pratas, A. J. Pinho, A. J. R. Neves, and C. A. C. Bastos, "DNA Synthetic Sequences Generated by Finite-Context Models," in *Proc. of the International Conference on Computational Advances in Bio and Medical Sciences (ICCABS)*, 2010.