EXPLORING LARGE LANGUAGE MODELS THROUGH N-GRAMS IMPLEMENTATION

Dang Dinh NGUYEN

Faculté des Sciences, Master IAAA, Aix - Marseille Université

ABSTRACT

N-gram models are a fundamental concept in language modeling, and our work focuses on implementing and analyzing their characteristics. During this exploration, we encountered the challenge of out-of-vocabulary (OOV) words, a common issue in natural language processing. To address this, we examined smoothing techniques like Laplace smoothing and Kneser-Ney smoothing, which improve the model's ability to handle unseen words or contexts. We then applied these models to the task of authorship identification, evaluating their effectiveness in a practical setting. Our experiments have provided valuable insights into the roles, strengths, and challenges of using n-gram models in real-world applications.

Keywords— Large Language Model (LLM), N-grams models, Markovian models, Natural Language Processing

1. INTRODUCTION

Although more advanced techniques have largely supplanted them, n-gram models remain a cornerstone in understanding the evolution of language modeling. Mastering the principles of n-gram models is essential for grasping the mechanisms underlying modern large language models (LLMs), as they shed light on fundamental challenges in modeling linguistic sequences and the innovative solutions developed to overcome them.

This work seeks to deepen our understanding of this foundational modeling approach, and its key components are summarized as follows:

- We begin by implementing and analyzing various n-gram language models to explore their characteristics, strengths, and limitations. This examination provides a solid foundation for understanding the basic principles of language modeling.
- One of the critical challenges in language modeling is the Out-of-Vocabulary (OOV) problem, where the inability to handle unseen words significantly impacts performance. To address this, we employ smoothing techniques such as add-one (Laplace) smoothing and Kneser-Ney smoothing to adjust probability distributions and enhance model robustness.
- Finally, we explore the practical application of n-gram models in authorship identification, aiming to demonstrate their effectiveness and relevance in real-world tasks.

In this report, we begin with an overview and an analysis of the implemented n-gram language models (Section 2). Next, we examine the impact of smoothing techniques on improving model robustness and accuracy (Section 3). This is followed by an exploration of authorship identification using n-gram models (Section 4). We then provide a discussion of our findings and potential future perspectives (Section 5), before concluding the report with a summary of key insights and contributions (Section 6).

2. AN IMPLEMENTATION OF N-GRAMS MODELS

2.1. Models

A n-gram language model is a purely statistical model of language. It is based on an assumption that the probability of the next word in a sequence depends only on a fixed size window of previous words. Mathematically, the probability of a sequence $S = w_1, w_2, ...w_T$ is approximated as:

$$P(S) = \prod_{t=1}^{T} P(w_t | w_{t-(n-1)}, ..., w_{t-1})$$

In our work, we implemented a range of n-gram language models with n varying from 1 to 3, corresponding to unigram, bigram, and trigram models. Each of these models was trained on a segmented corpus derived from the works of Alexandre Dumas (which consists of five novels split into training and test sets). During the training phase, each model estimated the probabilities of word sequences based on the distribution of words and their contexts within the training data, depending on the chosen n-gram configuration and estimation method.

2.2. Evaluation Metrics

2.2.1. The perplexity

Perplexity is a widely used metric for evaluating language models by assessing how well they predict a given text. It

is calculated as the exponential of the average negative log-likelihood of a sequence of words, with lower perplexity indicating better performance due to higher probabilities assigned to the observed words. Intuitively, perplexity measures the model's uncertainty: a perplexity of N implies the model is as uncertain as if it were choosing among N equally likely options for each word. To compare the performance of two models on a given corpus T, we simply compare their respective perplexity scores.

2.2.2. Missing word prediction

Another metric for evaluating the models is the accuracy achieved in predicting missing words within sentences. This metric assesses the model's ability to correctly infer and insert the appropriate word based on the given context, highlighting its understanding of linguistic patterns and dependencies.

2.2.3. Text generation

Lastly, we evaluated the model's ability to generate sentences on its own, testing its capacity to produce coherent and contextually appropriate sequences of words. This assessment helps gauge the model's generative capabilities and its understanding of language structure.

2.3. Results

This section presents the evaluation outcomes of the implemented n-gram language models, focusing on their predictive performance and practical applications. We analyzed the models across multiple metrics to assess their strengths and limitations. Below, we detail the performance of our models under different experimental conditions and highlight key observations.

2.3.1. Perplexities obtained by implemented models

We analyzed the perplexity scores of various models to evaluate their ability to capture text patterns. On the training corpus (Table 1), models employing the Maximum Likelihood Estimator (MLE) consistently outperformed those utilizing the Laplace Estimator. This result is expected, as the training process focuses on capturing n-grams and calculating probabilities without encountering unseen or low-frequency n-grams, thereby reducing the need for smoothing techniques. The reliable perplexities obtained with models implementing MLE present their strong capacity to effectively learn and represent patterns within a text.

To evaluate the generalization capability of the trained models, we tested them on separate test corpus. During testing, the models iterated through the test, identifying n-grams and attempting to associate each with the probabilities calculated during training. This process allowed us to determine how effectively the models could apply their learned patterns to unseen sequences and measure their performance beyond the training data.

	Perplexity (MLE)		
Corpus	Unigram	Bigram	Trigram
La_Reine_Margot	273,55	21,25	4,41
Le_comte_de_Monte_Cristo	362,8	24,16	4,12
Le_Vicomte_de_Bragelonne	273,33	25,85	5,79
Les_trois_Mousquetaires	298,41	23,2	4,61
Vingt_ans_apres	277,33	22,28	4,59

(a) Perplexities obtained by n-grams models which use ML estimator

	Perplexity (Laplace estimator)		
Corpus	Unigram	Bigram	Trigram
La_Reine_Margot	275,63	294,16	857,2
Le_comte_de_Monte_Cristo	366,35	491,17	1399,26
Le_Vicomte_de_Bragelonne	274,5	263,32	919,45
Les_trois_Mousquetaires	300,7	341,1	1042.43
Vingt_ans_apres	279,28	296,5	891,34

(b) Perplexities obtained by n-grams models which use Laplace estimator

Table 1: Perplexity scores of n-gram models with various probability estimation methods on train corpus

	Perplexity (MLE)		
Corpus	Unigram	Bigram	Trigram
La_Reine_Margot	inf	inf	inf
Le_comte_de_Monte_Cristo	inf	inf	inf
Les_trois_Mousquetaires	inf	inf	inf

(a) Perplexities obtained by n-grams models which use ML estimator

	Perplexity (Laplace estimator)		
Corpus	Unigram	Bigram	Trigram
La_Reine_Margot	289.71	365.03	1268.98
Le_comte_de_Monte_Cristo	378.14	607.85	2085.48
Les_trois_Mousquetaires	321.17	432.11	1581.00

(b) Perplexities obtained by n-grams models which use Laplace estimator

Table 2: Perplexity scores of n-gram models with various probability estimation methods on test corpus

Table 2 shows the results from our test phase. Models using Maximum Likelihood Estimator (MLE) produced infinite perplexity scores, indicating that MLE struggled with unseen words, as it assigns zero probability to any unobserved tokens in the training data. In contrast, models using Laplace smoothing yielded finite but high perplexity scores. Laplace smoothing prevents zero probabilities by adding a constant to all counts, but it can lead to over-smoothing, reducing the accuracy of probability estimates and increasing perplexity.

In conclusion, MLE models fail to generalize well to

unseen words, while Laplace smoothing helps but doesn't fully solve the issue. Given the challenge of handling Out-of-Vocabulary (OOV), we will explore alternative techniques in Section 3 to address this problem more effectively.

2.3.2. Missing word prediction

We evaluate the models' ability to predict a missing word using its context (the preceding words). Unigram models are excluded from this experiment, as they disregard the context and simply predict the most frequent word, resulting in nearly zero accuracy. In this evaluation, the models are trained on a corpus and tested on a set of 100 sentences from the same corpus, each with one word intentionally removed. The table below presents the accuracy achieved by our models in the task of predicting missing words:

Test Corpus	N-grams	Estimator	Prev	Prev+Succ
La_Reine_Margot.test.tok.100.1	2	MLE	0.31	0.7
		Laplace	0.31	0.44
	3	MLE	0.6	0.95
		Laplace	0.6	0.7
La_Reine_Margot.test.tok.100.2	2	MLE	0.22	0.68
		Laplace	0.22	0.34
	3	MLE	0.59	0.89
		Laplace	0.59	0.69
La_Reine_Margot.test.tok.100.3	2	MLE	0.18	0.54
		Laplace	0.18	0.33
	3	MLE	0.51	0.81
		Laplace	0.51	0.59

Table 3: Accuracy of n-gram models in predicting missing words

Initially, the prediction is made by selecting the word with the highest probability conditioned on the given context. This can be expressed mathematically as:

$$\hat{w_t} = \underset{w \in V}{\operatorname{argmax}} P(w_{t-(n-1)}, ..., w_{t-1})$$

The results indicate that the accuracies achieved by our models are relatively low, ranging from 30% to 60%. Trigram models consistently outperform bigram models, demonstrating notably higher accuracy scores. This pattern suggests that incorporating a larger context (two preceding words in the case of trigrams) provides the models with more information to make accurate predictions, which highlights the importance of context length in n-gram models.

We also observed no significant differences in performance between the two probability estimators, Maximum Likelihood Estimation (MLE) and Laplace smoothing. This indicates that smoothing techniques may not substantially enhance the ability of n-gram models to predict missing words when the available context is already limited.

To verify the hypothesis, we enhance the prediction process by incorporating both the preceding and succeeding contexts of the missing word. This approach is represented mathematically as:

$$\hat{w_t} = \underset{w \in V}{\operatorname{argmax}} P(w_{t-(n-1)}, ..., w_{t-1}) \cdot P(w_{t+1}, ..., w_{t+n})$$

We observed a significant increase in the accuracies of the models when incorporating both preceding and succeeding contexts. Models using MLE showed a substantial improvement of approximately 30%, while those using Laplace smoothing exhibited a more modest increase of around 10%. The trigram models still outperformed bigram models.

These patterns confirmed our hypothesis that the additional context provided by succeeding words plays a crucial role in improving the prediction of missing words, particularly for models that rely heavily on exact probability estimates, such as those using MLE. However, the limited improvement in models with Laplace smoothing indicates that it does not fully capture the complexities of broader contexts. This underscores the importance of incorporating richer contextual information and using advanced smoothing or modeling techniques to further enhance performance.

2.3.3. Text generation

To evaluate the performance of our models in text generation, we employed trigram models using Maximum Likelihood Estimation (MLE). This approach builds on prior findings that demonstrate the model's capability to effectively leverage contextual information.

The sentences generated by the model, as presented in Appendix A.2, reveal some promising attempts at sentence construction but also expose significant issues related to coherence, grammar, and overall clarity. Although the model successfully adheres to certain syntactic structures, the output often lacks logical consistency, with some sentences appearing fragmented or incomplete. For example, phrases such as "le comte de la compassion que de donner une vigueur peu commune" and "me ferez chasser" show contextual inconsistencies, while various grammatical errors, such as missing articles, incorrect verb conjugations, and unexpected word choices, are prevalent. Additionally, the model tends to repeat words, such as "d'Artagnan" or "dit-il," and occasionally generates unnatural phrasing, like "crie fais reconnaîtrai mains," which results in disordered word structures. These challenges suggest that, while the model demonstrates some ability to generate text, it faces difficulties in maintaining both syntactic and semantic coherence. This highlights the need for improvements in the training methodology, model architecture, or data quality to achieve more fluent and contextually accurate text generation.

3. KNESER-NEY SMOOTHING

The results presented in the previous section highlight one of the most critical challenges in language modeling: the

out-of-vocabulary (OOV) problem. This issue arises when the model encounters words during testing that were not observed in the training corpus, resulting in zero probabilities and thus infinite perplexity score. Basic solutions, such as Laplace smoothing, while mitigating the problem to some extent, have proven insufficient as they allocate probability mass uniformly across all unseen token, leading to a disproportionate reduction in predictive accuracy.

To address these challenges, advanced techniques like Kneser-Ney. In this section, we present the results of implementing Kneser-Ney smoothing on an n-gram model, analyzing its impact in comparison to MLE and Laplace smoothing. This analysis highlights how Kneser-Ney smoothing improves model robustness, particularly in scenarios with sparse data, and better manages the OOV problem by leveraging more sophisticated probability estimation mechanisms.

3.1. Description [1]

Kneser-Ney smoothing is a technique that refines the distribution of probability mass by leveraging both observed frequencies and the variety of contexts in which words appear. The core mathematical framework of Kneser-Ney smoothing for an n-gram model is built on three fundamental components: discounting, backoff, and continuation probabilities. The smoothed probability is formulated as:

$$P_{KN}(w_t|w_{t-(n-1)}^{t-1}) = \frac{\max(C_{KN}(w_{t-(n-1)}^{t}) - d, 0)}{w_{t-(n-1)}^{t-1}} + \lambda(w_{t-(n-1)}^{t-1})P_{KN}(w_t|w_{t-(n-1)}^{t-1}) \quad (1)$$

- Discounting (d): This term subtracts a fixed or dynamically computed discount d from the raw count $C_{KN}(w_{t-(n-1)}^{t}$. Discounting ensures that some probability mass is reserved for unseen events, addressing the zero-probability issue.
- Backoff ($\lambda(w_{t-(n-1)}^{t-1}$): When a specific n-gram has low or zero counts, the backoff term redistributes the reserved probability mass to lower-order $(n-1)$-grams. The backoff weight λ ensures that the total probability remains normalized.
- Continuation Probability: In contrast to traditional frequency-based estimators, Kneser-Ney smoothing uses continuation probabilities for lower-order $(n-1)$-grams. This accounts for the diversity of contexts in which a word appears, giving higher weight to words that occur in many different contexts, even if their raw frequency is low.

The recursive nature of Kneser-Ney smoothing lies in its reliance on lower-order $(n-1)$ -gram probabilities to estimate higher-order n-grams. The process continues until the unigram probabilities are reached, ensuring that all orders of n-grams contribute to the final probability estimate. This recursive structure is a key strength, as it balances local context (higher-order n-grams) with general patterns (lower-order n-grams) for robust probability estimation. These characteristics make it particularly effective for addressing sparse data and unseen word combinations in language modeling tasks.

3.2. Results

	Perplexity (Kneser-Ney)		
Train Corpus	Unigram	Bigram	Trigram
La_Reine_Margot	275.63	29.20	8.65
Le_comte_de_Monte_Cristo	366.34	35.84	9.16
Le_Vicomte_de_Bragelonne	274.50	32.73	10.29
Les_trois_Mousquetaires	300.69	32.08	9.30
Vingt_ans_apres	279.28	30.28	8.92

(a) Perplexities obtained by n-grams models with Kneser-Ney smoothing on train texts

	Perplexity (Kneser-Ney)		
Test Corpus	Unigram	Bigram	Trigram
La_Reine_Margot	289.71	56.47	40.16
Le_comte_de_Monte_Cristo	378.14	78.02	57.26
Le_Vicomte_de_Bragelonne	278.83	51.55	35.45
Les_trois_Mousquetaires	321.17	62.62	44.92
Vingt_ans_apres	297.75	56.73	39.82

(b) Perplexities obtained by n-grams models with Kneser-Ney smoothing on test sets

Table 4: Perplexity scores of n-gram models with Kneyser-Ney smoothing

The table above presents the perplexity results for models utilizing Kneser-Ney smoothing. Notably, the perplexities on the training sets for these models are slightly higher than those observed with MLE while remain significantly lower than those of models using Laplace smoothing.

On the test sets, the models using the Kneser-Ney estimator demonstrate superior robustness, achieving markedly better results than those with Laplace smoothing. For instance, on La Reine Margot, the perplexity drops dramatically from 1268.98 with Laplace smoothing to just 40.16 with Kneser-Ney smoothing—an improvement of approximately 90%.

This pattern suggests that Kneser-Ney smoothing not only mitigates the limitations of Laplace smoothing but also significantly enhances the model's ability to generalize across unseen contexts. These results reaffirm the importance of advanced smoothing techniques in developing robust language models that are capable of performing effectively in real-world applications.

However, indicate that this robustness does not extend to other tasks, such as missing word prediction or text genera-

tion. While Kneser-Ney smoothing effectively addresses perplexity and improves model generalization on unseen data, it does not directly enhance the semantic coherence or syntactic accuracy required for these tasks. For missing word prediction, the model's reliance on continuation probabilities may limit its ability to leverage richer contextual information effectively. Similarly, in text generation, the recursive backoff mechanism can lead to outputs that, while statistically plausible, lack the fluency and logical consistency expected in human-like text.

Another point worth noting is that when encountering a completely new word — one that has never appeared in any context—Kneser-Ney smoothing still fails to assign a non-zero probability. Consequently, the perplexity remains undefined or exceedingly high in such cases. This limitation highlights a critical weakness of n-gram models: their inability to handle out-of-vocabulary (OOV) words effectively. While smoothing techniques like Kneser-Ney redistribute probability mass to improve robustness, they inherently depend on the presence of some historical context within the training data. This underscores the need for more sophisticated models, such as neural network-based language models, which can generalize better to novel inputs by leveraging distributed representations and richer contextual embeddings.

4. AUTHORSHIP IDENTIFICATION

Authorship identification is a fascinating and challenging task in the field of computational linguistics, where the objective is to attribute a piece of text to its correct author based on distinctive stylistic features. This problem has practical applications in areas such as literary analysis, forensic linguistics, and historical studies, where the goal is to determine the authorship of anonymous or disputed texts. One widely used approach to solving this problem is the application of n-gram models, which analyze sequences of words (or characters) to capture the unique linguistic patterns of an author's writing style.

By training n-gram models on the writings of known authors, we can create a statistical representation of their style, focusing on word choice, syntax, and phrasing preferences. When presented with an unknown text, these models can compare its n-gram distribution to those of potential authors, making it possible to identify the most likely writer.

4.1. Dataset

To conduct this experiment, we collected works from approximately twenty popular authors available through the Project Gutenberg database [2]. While many of the selected authors originally wrote their novels in languages other than English, we specifically chose those works that had been translated into English. This approach was taken to ensure consistency across the corpus, allowing for a more reliable evaluation of the models. The works were downloaded in .html format and processed using a tool to remove extraneous content such as watermarks, titles, chapter headings, and other metadata, ensuring that only the core text remained. The text was then tokenized and split into training and test sets, with 30% of each author's work allocated to the test set (A comprehensive list of the works included in this experiment can be found in appendix - Table 6) . This cleaned and preprocessed dataset offers a variety of writing styles, making it an excellent resource for testing n-gram models in the task of authorship identification.

4.2. Methodologies

In this experiment, the task of author verification is approached as a classification problem, where the goal is to determine the author of a given text based on the distinctive characteristics of their writing style. To capture these stylistic patterns, n-gram models are employed, with a separate model trained for each author in the dataset. The perplexity values derived from these models are then used to assess the likelihood that a new, unseen text belongs to a specific author.

To verify the author of a text, we compute its perplexity under each trained n-gram model. This process involves calculating the smoothed probabilities of the n-grams in the text, applying Laplace and Kneser-Ney smoothing to handle the out-of-vocabulary (OOV) issue. The model that produces the lowest perplexity for the text is identified as the most likely author. Additionally, a list of the top 5 potential authors is generated based on their perplexity scores. This procedure is repeated for each text in the test set, allowing us to evaluate the model's performance in identifying the correct author.

Rather than relying on traditional classification metrics such as accuracy and confusion matrices, the performance of the verification method is assessed using alternative metrics: Top-k Accuracy and Mean Reciprocal Rank (MRR). These metrics are better suited for our task, as the models are not expected to make a single, binary prediction but instead rank multiple potential authors based on perplexity scores. Top-k Accuracy measures whether the true author is among the top k predicted authors, providing a more flexible and informative evaluation when the model is uncertain about the correct answer. Mean Reciprocal Rank (MRR) assesses how high the true author ranks, on average, across all test cases, offering insights into how well the model ranks the correct author compared to others. These evaluation metrics provide a more nuanced understanding of the model's performance in scenarios where the task involves ranking multiple candidates instead of making a definitive, single prediction.

4.3. Results

The experiment results show that the trigram model with Laplace smoothing, achieving a Top-5 Accuracy of 0.428

and a Mean Reciprocal Rank (MRR) of 0.166, still leaves considerable room for improvement. While the model correctly identifies the true author in some cases, it struggles to consistently rank the true author highly, as indicated by the low MRR and moderate Top-5 Accuracy.

In contrast, the trigram model with Kneser-Ney smoothing shows perfect performance, with a Top-5 Accuracy and MRR of 1.0. This indicates that Kneser-Ney smoothing effectively handles low-frequency n-grams and provides better probability estimates for unseen n-grams, leading to more precise predictions. However, the perfect performance may be influenced by the imbalance in corpus sizes, as models trained on larger datasets could dominate predictions, skewing results. Thus, while Kneser-Ney smoothing shows superior performance, the impact of dataset imbalance on generalization should be considered.

An interesting case emerged when testing two novels by Jack London, labeled JL1 and JL2. While the model correctly identified Jack London as the author of JL1, it did not rank other works by the same author (JL2) among the top 5 predictions. This suggests that the model may not capture the stylistic variations across different works by the same author. Factors such as genre, time period, or narrative voice could influence the writing style, causing the model to treat works from the same author as stylistically distinct. This highlights the model's limitation in generalizing across different texts by the same author, underscoring the need for improvements in handling intra-author stylistic variations.

5. DISCUSSION AND FUTURE WORK

The findings of this project underscore several inherent limitations in the implemented n-gram models, particularly in addressing the out-of-vocabulary (OOV) problem. While the application of Kneser-Ney smoothing effectively mitigated some challenges associated with unseen n-grams, it fell short in handling entirely novel words or rare combinations absent from the training data. Furthermore, the perplexity metric, though valuable in evaluating model performance, primarily captures probabilistic accuracy and does not adequately reflect a model's ability to generate semantically meaningful or syntactically accurate text. This limitation was evident in tasks such as missing word prediction and text generation, where contextual coherence and logical consistency were not fully achieved.

The scope of this work was intentionally constrained to fundamental n-gram models, and restricted within a controlled experimental setting, which may not fully generalize to more varied and ambiguous real-world issues.

Our future work would prioritize addressing these limitations by incorporating broader and more diverse corpora, encompassing various writing styles, genres, and languages. Advanced modeling approaches, such as neural network-based architectures, hold promise for overcoming the contextual and representational limitations of n-gram models. Additionally, we expect to expand the evaluation framework to include more complemented metrics for a more comprehensive assessment of model performance across diverse NLP tasks.

6. CONCLUSION

This project explored the implementation and evaluation of n-gram language models, emphasizing their foundational role in capturing local linguistic dependencies. While techniques like Kneser-Ney smoothing enhanced model robustness against sparse data, the inherent limitations of n-grams, such as their fixed context and inability to handle complex linguistic structures, were evident.

It also provided valuable insights into the challenges of language modeling and the trade-offs between simplicity and expressiveness. It reinforced our understanding of probabilistic methods and their role in shaping modern NLP techniques, while equipping me with the technical and analytical skills necessary for addressing more advanced modeling challenges in future.

7. REFERENCES

[1] Denny Ceccon, "A simple numerical example for kneser-ney smoothing [nlp]," *Medium*, 2019.

[2] "About Project Gutenberg," `https://www.gutenberg.org/about/`.

A. APPENDICES

A.1. Corpus Profiles and N-gram Distributions

	Corpus	Size	Vocabulary	Unigram	Bigram	Trigram
1	La_Reine_Margot.tok	328130	13088	13088	81938	166741
	La_Reine_Margot.train.tok	295314	12462	12462	75905	152345
	La_Reine_Margot.test.tok	32816	3844	3844	14434	22081
2	Le_comte_de_Monte_Cristo.tok	249494	13023	13023	76690	148522
	Le_comte_de_Monte_Cristo.train.tok	224538	12369	12369	71091	135709
	Le_comte_de_Monte_Cristo.test.tok	24956	3638	3638	12873	18659
3	Le_Vicomte_de_Bragelonne.tok	1092464	25303	25303	204046	473993
	Le_Vicomte_de_Bragelonne.train.tok	983196	24157	24157	189779	434292
	Le_Vicomte_de_Bragelonne.test.tok	109268	8285	8285	37856	65006
4	Les_trois_Mousquetaires.tok	348130	13984	13984	88761	183782
	Les_trois_Mousquetaires.train.tok	313274	13303	13303	82271	167976
	Les_trois_Mousquetaires.test.tok	34856	4122	4122	15884	24361
5	Vingt_ans_apres.tok	249494	392166	14472	94603	196444
	Vingt_ans_apres.train.tok	353022	13770	13770	87760	179540
	Vingt_ans_apres.test.tok	39144	4303	4303	16943	26206

Table 5: Corpus Statistics: Vocabulary and N-gram Counts

A.2. Generated sentences

$< s >$ traversant suis ami saint je revu et ! en a tiré d' affaire , et que nous traversons la tyne
$< s >$ dit - il , tout frais ! $< /s >$
$< s >$ j' ai fait entendre , ce me semble . $< /s >$
$< s >$ c' est le moine escortait le blessé sourit tristement et secoua la tête appuyée à la rivière . $< /s >$
$< s >$ le comte de la compassion que de donner une vigueur peu commune ; ses lèvres de carmin , comparées par
$< s >$ oui , dit porthos en se reculant , je ne l' ai coupé dans les rues saint - germain .
$< s >$ comme je ne me laisserez - vous de cet amour prévenant qu' un amant a pour sa famille . $< /s >$
$< s >$ — allons , maître d' hôtel ; et à mon gardien , celui sur lequel on commençait , de ne
$< s >$ — au fond , qui pendait jusqu' à la terre ; il avait joué . $< /s >$
$< s >$ me ferez chasser . $< /s >$
$< s >$ crie fais reconnaîtrai mains $< /s >$
$< s >$ d' artagnan , la reine . $< /s >$
$< s >$ d' artagnan fut le premier bruit qu' athos n' avait pas oublié ma bourse que j' ai tout vu ,
$< s >$ était - il . $< /s >$
$< s >$ — vous êtes couvert de sueur qui mouillait son visage un peu forcées et le cœur d' athos , dit
$< s >$ que voulez - vous ! $< /s >$
$< s >$ cela lui convient . $< /s >$
$< s >$ l' adresse d' athos ! $< /s >$
$< s >$ monsieur , surtout quand ces supérieurs sont princes , ce ne peut pas , et isaac était son propre compte
$< s >$ non , à nous . $< /s >$

A.3. Corpora Used in Authorship Identification

Author	Title	Size	Vocabulary
Alexandre Dumas	The Count of Monte Cristo	658590	21718
James M. Barrie	Peter Pan	69475	5440
Bram Stocker	Dracula	223983	11617
Cervantes	Don Quixote, Vol. I	254053	13231
Charles Dickens	Oliver Twist	231628	12898
Conan Doyle	Sherlock Holmes	151745	8717
Dostoevsky	White Nights and Other Stories	176070	10134
Scott Fitzgerald	The Great Gatsby	71875	6490
Frank Baum	The Wonderful Wizard of Oz	54569	3254
The Brothers Grimm	Grimms' Fairy Tales	127563	5979
Homer	The Odyssey	150196	7905
Jack London	The Call of the Wild	43616	4918
	Before Adam	55604	4530
Jane Austen	Pride and prejudice	174318	8949
Jules Verne	Around the World in Eighty Days	91223	7796
Lewis Carroll	Alice's Adventures in Wonderland	38342	3251
Mark Twain	The Adventures of Tom Sawyer	104205	8802
A. A. Milne	Winnie-the-pooh	36377	2226
Oscar Wilde	The Picture of Dorian Gray	120396	7566
Leo Tolstoy	War and Peace	798328	22677

Table 6: List of Authors and Texts

A.4. Top-K accuracy and Mean Reciprocal Rank (MRR)

Top-k Accuracy measures how often the true label appears within the top k predicted labels. This is useful when the model generates a ranked list of predictions, rather than just one. This metric can be illustrated as :

Top-K-Accuracy = (Number of correct predictions in the top-k) / (Total number of test samples)

Mean Reciprocal Rank (MRR) is used to evaluate the effectiveness of a ranked list of predictions. It calculates the reciprocal rank of the first correct answer for each query (test example). The reciprocal rank is defined as:

Reciprocal Rank = 1/(Rank of the first correct answer)