BOOKCOREF: Coreference Resolution at Book Scale

Giuliano Martinelli*, **Tommaso Bonomo***, **Pere-Lluís Huguet Cabot**, and **Roberto Navigli**
Sapienza NLP Group, Sapienza University of Rome
{martinelli, bonomo, huguetcabot, navigli}@diag.uniroma1.it

Abstract

Coreference Resolution systems are typically evaluated on benchmarks containing small- to medium-scale documents. When it comes to evaluating long texts, however, existing benchmarks, such as LitBank, remain limited in length and do not adequately assess system capabilities at the book scale, i.e., when co-referring mentions span hundreds of thousands of tokens. To fill this gap, we first put forward a novel automatic pipeline that produces high-quality Coreference Resolution annotations on full narrative texts. Then, we adopt this pipeline to create the first book-scale coreference benchmark, BOOKCOREF, with an average document length of more than 200,000 tokens. We carry out a series of experiments showing the robustness of our automatic procedure and demonstrating the value of our resource, which enables current long-document coreference systems to gain up to +20 CoNLL-F1 points when evaluated on full books. Moreover, we report on the new challenges introduced by this unprecedented book-scale setting, highlighting that current models fail to deliver the same performance they achieve on smaller documents. We release our data and code to encourage research and development of new book-scale Coreference Resolution systems at https://github.com/sapienzanlp/bookcoref.

1 Introduction

Coreference Resolution (CR) aims to identify and group mentions that refer to the same entity (Karttunen, 1969). Co-referring mentions are usually found across sentences and can be located far apart within the same document. For this reason, as the document length increases, so does the difficulty of manually annotating a corpus with coreference relations (Roesiger et al., 2018). This is because humans typically resolve coreference relations incrementally (Altmann and Steedman, 1988), and therefore – when annotating a coreference mention – they must rely on previously annotated entities and take the preceding context into account (Roesiger et al., 2018). This labor-intensive process results in a higher cost of human annotations of long-form texts. As a result, current benchmarks ease the annotation process by drawing samples from short- or medium-sized textual genres (i.e., news, broadcasts, and magazines), or by artificially shortening documents. For instance, the two most widely used English coreference benchmarks, OntoNotes (Pradhan et al., 2013) and LitBank (Bamman et al., 2020), either split documents into smaller chunks or truncate them to the first 2,000 tokens. For this reason, most CR systems are optimized for shorter

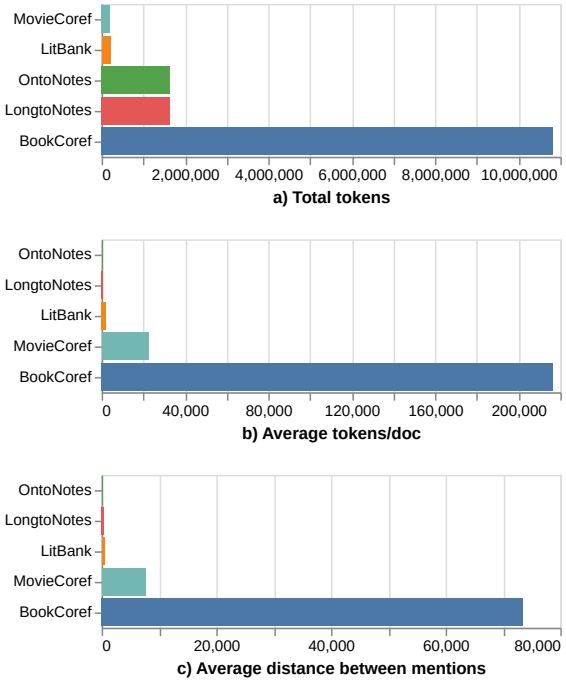

Figure 1: Comparison between BOOKCOREF and current long-document resources, measuring: a) the total number of tokens, b) the average number of tokens per document, and c) the micro-average pairwise distance between all co-referring mentions.

*Equal contribution.

texts and struggle to handle longer inputs effectively.

Several recent works, such as LongtoNotes (Shridhar et al., 2023) and MovieCoref (Baruah et al., 2021; Baruah and Narayanan, 2023), attempt to address this gap by introducing manually-annotated long-document CR resources. However, LongtoNotes documents remain limited in length (less than 700 tokens per document), while MovieCoref contains only a small number of full-length screenplays, a niche narrative genre with an atypical textual structure. We argue that the current lack of book-scale benchmarks leaves the many challenges this setting poses – such as efficient processing (Toshniwal et al., 2020, 2021), consistency (Guo et al., 2023), and evaluation (Duron-Tejedor et al., 2023) of Coreference Resolution in long documents – largely unexplored.

To address this limitation, we introduce a highly reliable automatic pipeline to annotate long documents, which we then apply to full-length literary works to create the first book-scale coreference benchmark, BOOKCOREF. Inspired by recent studies that suggest focusing on a smaller yet relevant set of entities (Baruah et al., 2021; Guo et al., 2023; Manikantan et al., 2024), our annotation involves only the characters that appear in a book, as they are the main agents of fictional stories (Bamman et al., 2013; Roesiger et al., 2018; Labatut and Bost, 2019). As shown in Figure 1, BOOKCOREF presents unprecedented long-document characteristics, enabling the study of the coreference phenomena in full narrative books, which was previously impossible. To summarize, in this work:

- We put forward the BookCoref Pipeline, a novel procedure that enables Coreference Resolution annotation of full documents, and extensively validate its effectiveness.
- We introduce BOOKCOREF, a new book-scale Coreference Resolution dataset with a manually-annotated split, BOOKCOREF$_{gold}$, to train and evaluate systems on fully annotated books.[1]
- We benchmark state-of-the-art Coreference Resolution systems, reporting on the open challenges of this new book-scale setting.

[1]We release BOOKCOREF on HuggingFace: https://huggingface.co/datasets/sapienzanlp/bookcoref.

2 Related Work

We now review established resources and approaches that deal with long-document Coreference Resolution. In Section 2.1, we delve into the details of the available datasets, underscoring the current lack of training and evaluation resources at the book scale. Subsequently, in Section 2.2, we survey recent state-of-the-art CR systems, highlighting that many are not well suited for processing long documents.

2.1 Long-document Benchmarks

Existing English Coreference Resolution benchmarks typically feature short- to medium-sized documents. Among these, OntoNotes (Pradhan et al., 2013) is the most widely used dataset, comprising documents from various genres, such as news, broadcast, and magazines, with an average length of 467 tokens. This brevity arises from splitting source documents into smaller partitions to simplify annotation. Shridhar et al. (2023) manually merge coreference clusters across partitions to construct full-document annotations (LongtoNotes), but, due to the short nature of the source genre, the average length only increases to 679 tokens.

WikiCoref (Ghaddar and Langlais, 2016), instead, explores the encyclopedic genre, proposing a 37-document evaluation set of Wikipedia pages with an average length of 1,995 tokens. Similarly, Bamman et al. (2020) introduce LitBank, an annotated benchmark of 100 book samples from the literature genre, where documents are long, and relatively few major entities play a central role. However, while LitBank has become the long-document standard for CR, it truncates book samples to 2,000 tokens, failing to capture coreference relations that develop across entire books.

Recent work explores the Coreference Resolution task in longer contexts. MovieCoref (Baruah et al., 2021; Baruah and Narayanan, 2023) consists of 6 full-sized screenplays and 3 excerpts, each manually annotated with coreference relations, reaching an average text length of 20,000 tokens. However, the number of annotated documents is relatively small, and their coreference annotations depend heavily on the underlying screenplay structure, which is very different from the free-form text found in other coreference datasets. Finally, Guo et al. (2023) manually annotate *Animal Farm* by George Orwell as a benchmark to test the capabilities of CR systems when applied to

longer text. Notably, both the annotation guidelines of MovieCoref and *Animal Farm* focus exclusively on characters, reflecting their central role in modern narrative analysis (Piper et al., 2021; Manikantan et al., 2024). In BOOKCOREF, we follow the same design choice and build a large-scale dataset, which we then use to train and evaluate CR capabilities of systems on a diverse set of narrative books.

2.2 Long-document Coreference Systems

Most recent state-of-the-art solutions for CR are specifically tailored to short- or medium-length documents and are not suited to the book-scale setting. Generative approaches that formulate CR as a sequence-to-sequence task (Bohnet et al., 2023; Zhang et al., 2023) usually require re-generating the entirety of the input text along with the coreference annotations, effectively doubling the context length. This is impractical for book-scale settings, as generative systems usually rely on very large Transformers, which have fixed input lengths and imply a high computational cost that becomes prohibitive when processing entire books. The same concerns apply to Large Language Models (LLMs), whose application to CR is still under discussion: current methods for LLM-based CR have yet to reach the performance of supervised models (Le and Ritter, 2024; Porada and Cheung, 2024).

Discriminative encoder-only models are in general more memory- and time-efficient (Otmazgin et al., 2022) but suffer from a similar limitation: recent approaches such as LingMess (Otmazgin et al., 2023) or s2e-coref (Kirstain et al., 2021) are constrained to respect the maximum input length of their underlying Transformer encoder, i.e., LongFormer (Beltagy et al., 2020). On the other hand, some recent encoder-only solutions can handle longer documents: the current state-of-the-art coreference resolution system, Maverick (Martinelli et al., 2024a), is built upon DeBERTa-v3 (He et al., 2023), which can encode up to 25,000 tokens (He et al., 2021). However, due to its quadratic computational complexity with respect to input length, processing entire books remains impractical, as hardware requirements scale rapidly.

To solve these issues, recent work proposes systems tailored to the processing of long text, employing an incremental formulation. Longdoc (Toshniwal et al., 2020, 2021) specifically tackles the runtime memory problems that long-document CR causes: it incrementally builds entity coreference clusters and learns to "forget" entities using a global cache of the most recent entities predicted. More recently, Dual cache (Guo et al., 2023) proposes modifying the Longdoc architecture by introducing a secondary cache that globally takes into account the least-frequently used mentions. As noted in Section 2.1, the researchers measure their improvements by annotating a full book, Orwell's *Animal Farm*. Interestingly, they show that their formulation outperforms other systems, but only reaches 36% CoNLL-F1, which is much lower than the average scores on other medium-sized datasets.

This highlights that i) a more complete book-level training and evaluation benchmark is needed to assess system performance on extended contexts, and that ii) current long-document systems cannot be used as automatic annotators, an issue we address through the BookCoref Pipeline.

3 BOOKCOREF

This Section describes the resources and methods we used to create our novel book-level benchmark, BOOKCOREF. Specifically, in Section 3.1 we introduce the underlying resources used to obtain the full texts and the character list of the books we annotate. Then, in Section 3.2, we present the BookCoref Pipeline, our automatic procedure for producing high-quality silver data for book-level CR. Finally, in Sections 3.3 and 3.4 we detail our manual annotation process and compare the resulting BOOKCOREF corpus with previous well-established CR benchmarks.

3.1 Underlying Resources

Our automatic pipeline annotates full-text documents starting from their list of characters. Therefore, to produce BOOKCOREF we leverage three main resources: i) Project Gutenberg[2], a collection of openly available literary works; ii) Wikidata, a multilingual knowledge graph containing thousands of books[3]; iii) LiSCU (Brahman et al., 2021), a dataset that collects information from online study guides on the characters appearing in English literary works.

We start by collecting a list of characters, authors, and titles of books available in Wikidata and LiSCU. As these resources do not provide full texts, we search Project Gutenberg for complete books that match the corresponding authors and titles. To ensure high quality, we manually remove erro-

[2] https://www.gutenberg.org/

[3] As of 2024-12-09, Wikidata contains 75,675 book items.

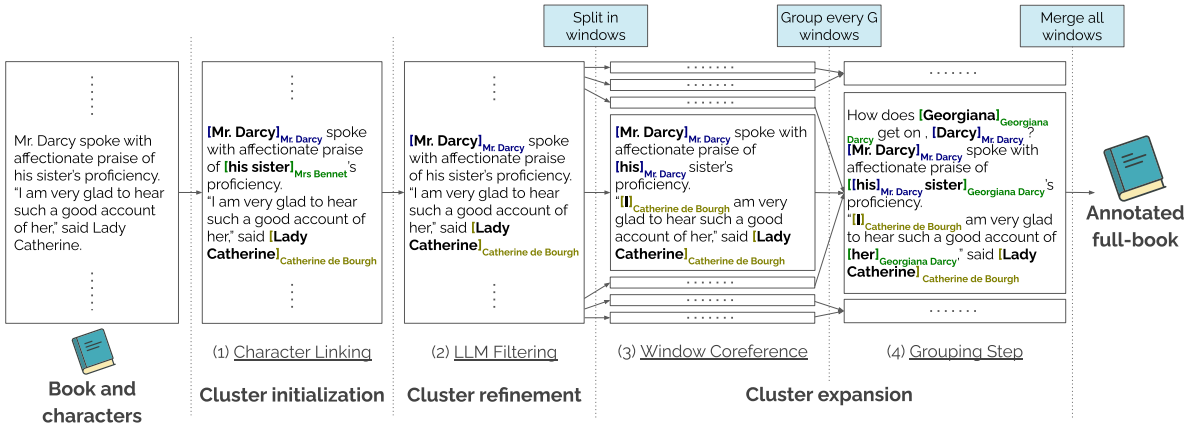

Figure 2: The BookCoref Pipeline applied on a sample taken from *Pride and Prejudice*. (1) Link all explicit character mentions via Character Linking. (2) Filter out inconsistent assignments via LLM Filtering. (3) Expand character clusters using a CR model on small windows. (4) Expand character clusters of grouped windows.

neous matches and automatically exclude books with fewer than five characters.

Formally, after this step, we obtain a set of full-text books $\mathcal{B}$ where each book $b \in \mathcal{B}$ is paired with a list of character names $\mathcal{C}(b) = \{c_1, c_2, \ldots, c_n\}$. Our corpus contains $|\mathcal{B}| = 53$ books with an average of 27 characters per book, mostly comprising full-length classical novels such as Walter Scott's *Ivanhoe* or Jane Austen's *Pride and Prejudice*. In Appendix A we provide a more detailed analysis of our corpus and preprocessing step, along with a detailed table of all the books included in BOOKCOREF.

3.2 BookCoref Pipeline

We now introduce the BookCoref Pipeline, our proposed automatic annotation procedure. Formally, the BookCoref Pipeline takes as input a book b and its set of character names $\mathcal{C}(b)$ and outputs coreference cluster annotations over the full book for each character name. Notably, although we apply our pipeline to narrative books, it is genre-agnostic and can be exploited for any long document, provided a set of named entities is available. Figure 2 summarizes the steps of our BookCoref Pipeline, namely, Cluster Initialization (Section 3.2.1), Cluster Refinement (Section 3.2.2), and Cluster Expansion (Section 3.2.3).

3.2.1 Cluster Initialization

In our first step, we initialize character coreference clusters[4] by tagging explicit entity mentions (i.e., non-pronoun mentions such as proper nouns or noun phrases). We extract these mentions through Entity Linking (EL), the task of linking explicit mentions of entities in a text to their corresponding entry in a Knowledge Base (KB).

In our case, the index (or KB) of possible entities changes for each book b and corresponds to the list of character names $\mathcal{C}(b) = \{c_1, c_2, \ldots, c_n\}$. Formally, given a character c from a book $b \in \mathcal{B}$, we initialize its coreference cluster $\mathcal{E}_c^b$ with all its explicit mentions extracted through an automatic EL system. We note as $\{\mathcal{E}_c^b\}_{c \in \mathcal{C}(b)} = \{\mathcal{E}_{c_1}^b, \mathcal{E}_{c_2}^b, \ldots, \mathcal{E}_{c_n}^b\}$ the set of all character coreference clusters in a book b.

The main limitation of current pre-trained EL systems is that they are trained to link named entities to predefined knowledge bases, such as Wikipedia or Wikidata; in our case, instead, we aim to link explicit mentions of characters to their respective names, a task that we define as Character Linking. To this end, we prepare a small training set for the Character Linking task. Specifically, we manually amend LitBank (Bamman et al., 2020) by filtering out all non-explicit character mentions and naming each cluster with character names (full details are provided in Appendix B). We then use this resource to fine-tune a state-of-the-art EL system (Orlando et al., 2024, ReLiK) for Character Linking in the narrative genre.

We apply this trained system to our collection of books $\mathcal{B}$, obtaining the aforementioned character coreference clusters $\{\mathcal{E}_c^b\}_{c \in \mathcal{C}(b)}$, for each book $b \in \mathcal{B}$. As a by-product, our clusters are linked to a specific name, which is crucial to the next steps of our pipeline.

[4]By character coreference cluster, we refer to the set of coreferential mentions that correspond to the same character.

3.2.2 Cluster Refinement

We place a strong emphasis on the precision of our initial cluster formation: false positives (i.e., incorrect mention-character links predicted by our system) can easily propagate through the whole pipeline, whilst false negatives (i.e., correct mention-character links that are not predicted by our system) are easier to recover in our subsequent Cluster Expansion step. Therefore, we introduce an additional verification step for each mention of a character cluster by prompting an LLM to ascertain whether the mention is accurately linked to the character, based on the surrounding context.

Specifically, we prompt Qwen2 7B Instruct (Yang et al., 2024)[5] with the name of the character and the highlighted mention in context, and we constrain the LLM to output either 'Yes' or 'No' as the answer to our prompt (see Appendix C for the exact prompt and further details). We filter out mentions for which the LLM answers negatively. The final result of this step is a set of LLM-validated coreferential clusters $\left\{\hat{\mathcal{E}}_c^b\right\}_{c\in\mathcal{C}(b)}$ for each character c of all our books $\mathcal{B}$.

3.2.3 Cluster Expansion

As a result of the previous steps, for each book we have a set of high-precision annotations of coreference clusters containing the explicit mentions of each character that appears in that particular book.

To fully tag our books with complete CR annotations, we expand each coreference cluster by including every missing mention of a given character, including pronouns, noun phrases, and other references not tagged in the previous steps. We employ an automatic CR system, Maverick (Martinelli et al., 2024a), because it combines state-of-the-art performance with the ability to complete partial coreferential clusters, which is a prerequisite in order to exploit the annotations obtained in the previous steps. However, the problem with using Maverick in a long-context setting is that it suffers from the same input length issues that impact other CR models, as discussed in Section 2.2. Therefore, for each book $b \in \mathcal{B}$, we apply Maverick to consecutive, non-overlapping windows $\mathcal{W}(b) = \{w_1, \ldots, w_N\}$, with each window w_i containing a maximum number of 1500 words, the closest setting to Maverick's training conditions. More precisely, for each window $w_i \in \mathcal{W}(b)$ in book b, we derive from the set of coreferential clusters produced in the previous steps $\left\{\hat{\mathcal{E}}_c^b\right\}_{c\in\mathcal{C}(b)}$ a local set of clusters $\left\{\hat{\mathcal{E}}_c^{w_i}\right\}_{c\in\mathcal{C}(b)}$ by filtering out all mentions of a cluster that are not contained in window w_i. Following this, we use Maverick on window w_i to expand each character cluster $\hat{\mathcal{E}}_c^{w_i}$ to all of its coreferential mentions, including pronouns, noun phrases and other references, obtaining a local and complete set of clusters $\{\mathcal{M}_c^{w_i}\}_{c\in\mathcal{C}(b)}$ for a window w_i of a book b.

To obtain a full-book annotation for b, we merge the annotations from all the windows $w_i \in \mathcal{W}(b)$. This can be done by taking the union of every cluster linked to the same character c across all windows: $\mathcal{M}_c^b = \bigcup_{i=1}^{N} \mathcal{M}_c^{w_i}$. We leverage the fact that local coreference clusters are each linked to the same global set of character names, as discussed in the previous section. Although merging all windows $w_i \in \mathcal{W}(b)$ results in a precise annotation of book b, this approach does present a particular edge case that affects the recall of character mentions: when a window w_i lacks explicit mentions of a character $c \in \mathcal{C}(b)$, our Cluster Expansion step becomes ineffective, as there are no mentions from which to expand and form the corresponding coreferential cluster. This is outlined in Figure 2, where the mention "his sister" is not linked to the character *Georgiana Darcy*.

To mitigate this limitation, we propose an intermediate grouping step in which we merge G consecutive windows into a single grouped window[6], and run a second expansion step on a context that is larger and contains more characters. We therefore introduce Maverick$_{xl}$ (Section 4.2), an adaptation of Maverick that processes longer documents by encoding them in smaller splits, and use it to perform this second Cluster Expansion step. Formally, we define non-overlapping, grouped windows $gw_j \in \{gw_1, \ldots, gw_M\}$, with $M = \lceil |\mathcal{W}(b)| \;/\; G \rceil$, as the union of G consecutive windows, $gw_j = \bigcup_{i=G\cdot(j-1)+1}^{\min(G\cdot j, N)} w_i$, where $w_i \in \mathcal{W}(b)$. For each gw_j, we define its set of character clusters as the union of the local coreference clusters of the grouped windows: $\left\{\mathcal{M}_c^{gw_j}\right\}_{c\in\mathcal{C}(b)} = \left\{\bigcup_{i=G\cdot(j-1)+1}^{\min(G\cdot j,N)} \mathcal{M}_c^{w_i}\right\}_{c\in\mathcal{C}(b)}$. This grouping step expands the character clusters by considering the broader context provided by the grouped windows.

[5] We chose Qwen2 7B due to its permissive Apache 2.0 license.

[6] By performing a statistical analysis on our corpus, we found that $G = 10$ results in an optimal intermediate length between the size of windows and full books.

Datasets	Docs	Tok.	Ment.	Tok./Doc	Ment./Doc	Chains/Doc	Ment./Chain	Ment. Dist.
OntoNotes	3,493	1.6M	194k	467	55	12.7	4.3	140
LongtoNotes	2,415	1.6M	194k	674	80	16.7	4.9	371
LitBank	100	210k	29k	2,105	291	79.3	4.1	480
WikiCoref	30	59k	7k	1,996	233	49.4	4.5	1,013
MovieCoref	9	201k	26k	22,423	2,865	46.4	89	7,672
BOOKCOREF$_{gold}$	3	229k	24k	76,419	7,844	22.3	359	34,880
BOOKCOREF$_{silver}$	50	10.8M	968k	216,626	19,471	27.4	870	73,432

Table 1: Statistics of BOOKCOREF compared with previous CR datasets. Columns from left to right: total number of documents, tokens, and mentions; average number of tokens, mentions, and coreference chains per document; micro-average number of mentions per chain; and micro-average pairwise distance between mentions of the same chain. We use (k) and (M) to indicate thousands and millions, respectively.

We apply Maverick$_{xl}$ to each grouped window gw_j starting from $\{\mathcal{M}_c^{gw_j}\}_{c\in\mathcal{C}(b)}$ and we obtain a local set of coreference resolution annotations $\{\mathcal{F}_c^{gw_j}\}_{c\in C(b)}$. The benefits of this step can be seen clearly in Figure 2, where the mention "his sister" is correctly linked to the character *Georgiana Darcy*. Finally, for each character cluster, we take the union across these grouped windows gw_j, obtaining coreference annotations $\{\mathcal{F}_c^b\}_{c\in C(b)}$ for each book, which altogether represent the final coreference annotation of BOOKCOREF$_{silver}$.

3.3 Manually-annotated Test Set

To enable a complete and rigorous evaluation of coreference resolution models at book scale, we put forward a manually annotated benchmark, BOOKCOREF$_{gold}$. We include the annotation provided by Guo et al. (2024) for *Animal Farm* by George Orwell and two new, fully-annotated books, i.e., Herman Hesse's *Siddhartha* and Jane Austen's *Pride and Prejudice*. This latter is a particularly challenging benchmark as previous work has shown that its high density of characters and pronominal expressions often leads CR systems to produce erroneous links (Vala et al., 2015).

Our annotation guidelines mirror the ones of *Animal Farm* and are further explained in Appendix D. Each annotator is presented with the full text of the book and the corresponding list of characters, and is tasked with tagging any mention of the characters by means of an annotation tool.[7] The annotation was conducted by three expert CR annotators (authors of this paper) for around 120 hours, covering a total of 194,280 words. Following recent literature, the agreement was computed on a 2,000-word subset using traditional coreference metrics, achieving a 96.1 inter-annotator MUC score – comparable to LitBank (95.5) and higher than OntoNotes (83.0). Notably, restricting annotations to characters contributes to a high agreement.

3.4 Statistics

In Table 1 we report the statistics of our two new resources, BOOKCOREF$_{silver}$ and BOOKCOREF$_{gold}$, compared to well-established long-document benchmarks. Some distinctly book-scale characteristics emerge, such as very long documents, a low number of highly populated chains for each document, and a high average pairwise distance between mentions of the same chain.[8] BOOKCOREF$_{silver}$ is both large-scale, with 6.75 times the number of tagged tokens and 5 times the amount of mentions compared to OntoNotes, and book-level, with mentions linked across unprecedented distances. Despite containing only three manually-annotated books, we note that BOOKCOREF$_{gold}$ contains 229k tokens, ~10 times more than the test splits of well-established long-document benchmarks, LitBank (~21k) and MovieCoref (~29k).

3.5 BookCoref Pipeline Evaluation

We measure the performance of each step of our BookCoref Pipeline on our manually curated subset of books, BOOKCOREF$_{gold}$. Table 2 reports traditional CR metrics and standard metrics for Character Linking. In particular, we measure MUC (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998), CEAF$_{\phi_4}$ (Luo, 2005) and their average value, CoNLL-F1, for Coreference Resolution, and precision, recall and F1-score for Character Linking. We evaluate our initial Cluster Initialization step performance, comparing our approach to a simple Pattern Matching (PM) baseline, where character clusters are initialized by selecting mentions that match the character name. Our Character

[7] https://github.com/nilsreiter/CorefAnnotator

[8] The average pairwise distance is calculated as the micro-average of all the distances, measured in number of tokens, between any two mentions that are part of the same chain.

	Character Linking			Coreference			
	P	R	F1	MUC	B^3	$CEAF_{\phi_4}$	CoNLL
Cluster Initialization							
Pattern matching	95.6	18.9	29.2	14.7	4.7	34.5	17.9
Character Linking (CL)	88.6	30.9	44.5	39.3	12.7	50.5	34.2
Cluster Refinement							
CL + LLM filtering	93.8	29.8	43.5	37.5	50.9	12.9	33.9
Cluster Expansion							
CL + Window Coreference	85.7	75.3	80.2	85.7	62.9	65.3	71.3
+ Grouping step	78.7	87.8	83.0	91.0	65.2	67.8	74.7
CL + LLM filtering + Window Coreference	90.3	80.4	84.7	85.6	67.0	78.6	77.7
+ Grouping step	83.2	89.8	86.3	93.3	70.8	77.5	80.5

Table 2: Evaluation of our BookCoref Pipeline on BOOKCOREF$_{gold}$. We report precision, recall and F1-score for Character Linking and measure MUC, B^3, $CEAF_{\phi_4}$ and CoNLL-F1 as coreference metrics.

Linking method obtains high precision scores and ensures a higher recall of character mentions compared to PM, resulting in an increase of +15.3 in F1-score. Applying LLM-based filtering on top of our Character Linking outputs increases the character precision by an average of +5.2 points, as it removes several wrongly-predicted mentions. Keeping these mentions would result in a propagation of annotation errors in the following Cluster Expansion step: in Appendix Table 7 we present a qualitative analysis of the impact of this step. Regarding the final Cluster Expansion step, we show that the intermediate grouping strategy improves coreference-specific metrics, reaching 80.5 CoNLL-F1 score, and therefore we adopt it in the final BookCoref Pipeline. Finally, we highlight that our silver-annotation pipeline achieves an MUC score of 93.3, comparable to the inter-annotator MUC agreement between our human annotators (96.1) and between the annotators of other datasets (LitBank 95.5, OntoNotes 83.0, cf. Section 3.3).

4 Experimental Setup

In this Section, we report the experimental setup we developed to empirically analyze the training and testing of current systems on BOOKCOREF.

4.1 Experiments Design

We benchmark automatic coreference systems on the proposed BOOKCOREF dataset. Specifically, we test the models both off-the-shelf, i.e., loading their pre-trained configurations, and after training them on BOOKCOREF$_{silver}$. We measure CR performance in two specific settings: i) on BOOKCOREF$_{gold}$, in which models are evaluated on book-scale coreference by taking in input full books; ii) on SPLIT-BOOKCOREF$_{gold}$, in which models are evaluated on medium-sized texts that result from splitting BOOKCOREF$_{gold}$ into independent windows of 1500 tokens. The two settings allow us to evaluate the ability of current CR systems to overcome the intrinsic difficulty of processing book-scale texts in comparison with medium-sized ones.

4.2 Comparison Systems

As discussed in Section 2.2, many available coreference systems cannot be applied to the book-scale setting. Different modeling approaches (LLMs, seq-to-seq, encoder-only) present significant limitations, which we describe in detail in Appendix E. Therefore, in our experiments, we benchmark available solutions for long documents and adapt a state-of-the-art coreference system to the book-scale setting. Among long-document systems, we benchmark the widely-adopted BookNLP library[9], a BERT-based model that is trained on LitBank. We also include Longdoc (Toshniwal et al., 2020, 2021) and Dual cache (Guo et al., 2024), two systems that were specifically designed to handle long texts through an incremental formulation of CR.

We compare the results of these systems with Maverick (Martinelli et al., 2024a), a non-incremental architecture that currently achieves state-of-the-art results on OntoNotes. To use Maverick on our book-scale setting, we adapt its architecture at inference time: instead of encoding documents in their entirety, we perform mention extraction on smaller windows of arbitrary length L. We then carry out the subsequent clustering steps on all the mentions of the books, drastically reducing the memory requirements that would be involved in encoding the whole book. We name this model Maverick$_{xl}$, and we set $L = 4000$, the largest value that allows us to run infer-

[9] https://github.com/booknlp/booknlp

	BookCoref$_{gold}$							Split-BookCoref$_{gold}$			
Models	Ani.	P.&P.	Siddh.	MUC	B^3	C_{ϕ_4}	CoNLL	MUC	B^3	C_{ϕ_4}	CoNLL
Off-the-shelf											
BookNLP	41.1	41.2	45.1	83.1	40.9	2.4	42.2	81.1	52.3	18.7	50.6
Longdoc	39.1	48.9	44.8	79.9	52.1	7.9	46.6	80.7	63.8	39.0	61.2
Dual cache	36.7	42.2	50.9	82.3	41.0	4.2	42.5	82.9	67.8	43.9	64.8
Maverick$_{xl}$	29.1	39.2	50.8	81.2	35.6	6.1	41.2	83.8	69.5	46.1	66.5
Fine-tuned on BookCoref$_{silver}$											
Longdoc	**67.5**	**63.9**	**74.0**	93.5	**62.4**	**45.3**	**67.0**	91.2	74.9	65.7	77.1
Dual cache	52.6	47.9	68.9	92.5	48.0	16.9	52.5	91.2	74.5	66.2	77.3
Maverick$_{xl}$	62.7	57.5	68.0	**94.3**	55.3	33.4	61.0	**92.7**	**82.2**	**71.9**	**82.2**

Table 3: Comparison between off-the-shelf models and systems trained on BookCoref$_{silver}$, tested on BookCoref$_{gold}$ and Split-BookCoref$_{gold}$. For each system, we report F1 measures of MUC, B^3 and CEAF$_{\phi_4}$ (noted as C_{ϕ_4}), and use their average, CoNLL-F1, as the main evaluation criteria. The first three columns detail avg. CoNLL-F1 scores on the three different books of our test-set, namely *Animal Farm*, *Pride and Prejudice* and *Siddharta*. We highlight in **bold** the best measures of trained models, and underline best off-the-shelf results.

ence on BookCoref$_{gold}$ on our academic-budget setup, i.e., a single NVIDIA RTX-4090 with 24GB of VRAM. Notably, Maverick and Maverick$_{xl}$ are equivalent when testing the model on Split-BookCoref$_{gold}$, where the input text is shorter than 4000 tokens.

With the exception of BookNLP[10], we train our comparison systems on BookCoref$_{silver}$. We split documents and respective coreference clusters into separate windows of a maximum length of 1500 tokens. This length both complies with the maximum input length specified by the adopted Transformer encoders, and also enables us to train all the comparison systems on our hardware avoiding the huge memory cost of training on full books[11].

We reserve Appendix F for further details on the comparison systems and our training setup.

5 Results

In Table 3, we benchmark current models on BookCoref$_{gold}$ and on Split-BookCoref$_{gold}$.

BookCoref$_{gold}$ Off-the-shelf models all achieve a CoNLL-F1 score above 40 points, with Longdoc obtaining the best performance of 46.6. Notably, in this setting, the popular BookNLP performs similarly to other neural counterparts.

As expected, training our comparison systems on the BookCoref$_{silver}$ data consistently improves their performance. In this setting, Longdoc is still the best-performing model, scoring 67.0 CoNLL-F1 points (+20.4 compared to the off-the-shelf version), demonstrating the superiority of its incremental formulation. Interestingly, Dual cache, which was previously considered the best choice for long-document CR, achieves low scores after fine-tuning, even when compared with Maverick$_{xl}$, a model that is not specifically tailored for long-document processing.

Our results demonstrate that none of the available off-the-shelf systems could have been considered a good candidate for the annotation of our silver corpus. In fact, the proposed BookCoref Pipeline obtains 80.5 CoNLL-F1 score on BookCoref$_{gold}$ (Table 2), +33.9 compared to the best off-the-shelf model.

Notably, by training on BookCoref$_{silver}$, Longdoc scores 67.5 CoNLL-F1 points on the Animal Farm benchmark, an increase of 31.2 points compared to previous literature (Guo et al., 2024). We reserve Appendix G for a more detailed report on the performance and robustness of the comparison systems.

Split-BookCoref$_{gold}$ Results on Split-BookCoref$_{gold}$ show an interesting finding: both off-the-shelf and fine-tuned systems perform much better in this medium-sized text setting, with Maverick$_{xl}$ reaching the best results. Specifically, Maverick$_{xl}$ scores 82.2 CoNLL-F1 points when trained on the BookCoref$_{silver}$ data, surpassing the second-best model, Dual cache, by +4.9 points.

We note that this result aligns with previous work, demonstrating that the incremental formulation of Longdoc is particularly robust on full books, whilst Maverick's single forward pass approach is superior on medium-sized text.

Metrics Comparing the measures of the full-book setting with the ones obtained with smaller

[10] BookNLP's codebase does not allow training on new datasets.

[11] For reference, training Maverick on a single book of 300k tokens requires more than 1200 GB of VRAM.

	Time (sec.)	Memory (GB)
Off-the-shelf		
BookNLP	180	6.9
Longdoc	**70**	**5.8**
Dual cache	460	11.6
Maverick$_{xl}$	211	14.8
Trained on BOOKCOREF$_{silver}$		
Longdoc	**26**	**5.8**
Dual cache	473	11.8
Maverick$_{xl}$	183	12.8

Table 4: Efficiency of each model when running inference on BOOKCOREF$_{gold}$. We measure the total execution time in seconds and the maximum GPU memory in gigabytes, marking the best scores in bold. All experiments were run on an NVIDIA RTX-4090.

windows unveils a recurring pattern: traditional coreference scores, i.e., CEAF$_{\phi_4}$ B^3 and MUC, show discrepancies within each comparison system when evaluating full-book performance.

Interestingly, MUC often measures over 80 F1 points both in the window and book-scale setting, while B^3 and, in particular, CEAF$_{\phi_4}$, often show lower scores on full books, which substantially increase when evaluating model performance on smaller texts.

While many previous works have shown weaknesses and disagreements of these metrics in specific scenarios (Moosavi and Strube, 2016; Borovikova et al., 2022; Duron-Tejedor et al., 2023), we argue that this new proposed book-scale setting offers fresh ground for improving the study of robustness and expressivity of standard CR metrics. We reserve Section G in the Appendix to obtain further insights into the behavior of coreference metrics and models in our book-scale scenario. Specifically, we propose an intermediate setting where full-book predictions are evaluated on smaller windows, and find that the discrepancy between CR metrics is lower in this setting compared to the original BOOKCOREF$_{gold}$. These additional results highlight the need to increase the robustness of CR metrics in the book setting.

Efficiency In Table 4 we detail the time and memory requirements of the comparison systems on our full book evaluation. Our results show that Longdoc achieves the lowest inference time and memory usage when processing BOOKCOREF$_{gold}$. We also note that Longdoc improves its time efficiency after being trained on BOOKCOREF$_{silver}$. We believe that the model learns to extract fewer, more precise mentions, which impacts its incremental clustering step and reduces processing time to 26 seconds.

Open Challenges of Book-scale CR Our results show that Longdoc achieves the best scores on book-scale coreference resolution after training on BOOKCOREF$_{silver}$.

We point out that the score of 67.0 CoNLL-F1 points on BOOKCOREF$_{gold}$ is relatively low compared to the score models attain on other CR benchmarks such as OntoNotes and LitBank, which is around 80 CoNLL-F1 points. Nevertheless, the results obtained on SPLIT-BOOKCOREF$_{gold}$ show that models can reach similar scores when dealing with smaller windows of text, attaining 82.2 CoNLL-F1 points. This demonstrates that the difference in performance between BOOKCOREF$_{gold}$ and SPLIT-BOOKCOREF$_{gold}$ is not due solely to the lack of manually annotated training resources. On the contrary, we argue that this difference highlights the open research challenges of our new book-scale setting, such as i) studying the interpretation of current coreference metrics for full-book evaluation, ii) creating new systems that can fully exploit book annotations, and iii) exploring efficient solutions for adapting current generative and encoder-only models for book-scale processing without incurring exponential computational costs.

6 Conclusion

In this paper, we fill the current gap in book-scale coreference resolution by proposing BOOKCOREF, a novel benchmark with unprecedented book-scale characteristics. We put forward BOOKCOREF$_{silver}$, a silver training corpus, and BOOKCOREF$_{gold}$, a manually annotated dataset for model evaluation. BOOKCOREF$_{silver}$ is annotated using the BookCoref Pipeline, a novel procedure that can provide full-text coreference annotations of the characters in a book, starting from the character names. We carefully validate our automatic approach intrinsically, measuring its effectiveness on our manually annotated dataset, and also extrinsically, showing that long-document systems benefit from training on our silver resource. Nevertheless, our results on BOOKCOREF$_{gold}$ show that specifically tailored long-document systems have a notable decrease in performance when evaluated on full books compared to that on smaller texts.

By releasing BOOKCOREF, we enable the investigation of the new research challenges posed by the book-scale setting, such as developing more accurate systems and studying long-document metrics.

7 Limitations

Although we measure and report the high-quality annotations of BOOKCOREF$_{silver}$, we remark that it is created automatically, and therefore may contain errors. The lack of manually annotated data for training may represent a possible limit in creating more robust book-scale CR systems and should be addressed in future annotation efforts.

Moreover, the coreference annotations we provide are limited to book characters. This decision is aligned with previous work on literary CR and is strongly motivated from a narrative perspective. We argue that, as a first foray into truly book-scale Coreference Resolution, this trade-off will still enable future research on this new setting.

We also note that our test set is composed of canonical texts, i.e., books that have been studied extensively and are readily available on the Internet. Testing with less renowned books would benefit the evaluation capabilities of BOOKCOREF$_{gold}$, a direction that should be prioritized by future work.

Furthermore, our implementation of the BookCoref Pipeline is currently focused on the English language, potentially limiting its broader applicability. However, we note that integrating multilingual automatic systems for Character Linking and Coreference Resolution into the BookCoref Pipeline would be enough to create multilingual corpora. We leave this intuition to future work.

Finally, we could not test some of the currently available systems on the book-scale setting. Our experiments were limited by hardware availability, i.e., a single RTX-4090. For this reason, we could not benchmark large generative models as we did with encoder-only systems.

Acknowledgements

We gratefully acknowledge the support of the PNRR MUR project PE0000013-FAIR.

Roberto Navigli also gratefully acknowledges the support of the CREATIVE project (CRoss-modal understanding and gEnerATIon of Visual and tExtual content), which is funded by the MUR Progetti di Rilevante Interesse Nazionale programme (PRIN 2020). This work has been carried out while Giuliano Martinelli was enrolled in the Italian National Doctorate on Artificial Intelligence run by Sapienza University of Rome.

References

Gerry Altmann and Mark Steedman. 1988. Interaction with context during human sentence processing. *Cognition*, 30(3):191–238.

Amit Bagga and Breck Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 79–85, Montreal, Quebec, Canada. Association for Computational Linguistics.

David Bamman, Olivia Lewke, and Anya Mansoor. 2020. An annotated dataset of coreference in English literature. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 44–54, Marseille, France. European Language Resources Association.

David Bamman, Brendan O'Connor, and Noah A. Smith. 2013. Learning latent personas of film characters. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 352–361, Sofia, Bulgaria. Association for Computational Linguistics.

Sabyasachee Baruah, Sandeep Nallan Chakravarthula, and Shrikanth Narayanan. 2021. Annotation and evaluation of coreference resolution in screenplays. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2004–2010, Online. Association for Computational Linguistics.

Sabyasachee Baruah and Shrikanth Narayanan. 2023. Character coreference resolution in movie screenplays. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10300–10313, Toronto, Canada. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The Long-Document Transformer. *CoRR*, abs/2004.05150.

Bernd Bohnet, Chris Alberti, and Michael Collins. 2023. Coreference resolution through a seq2seq transition-based system. *Transactions of the Association for Computational Linguistics*, 11:212–226.

Mariya Borovikova, Loïc Grobol, Anaïs Halftermeyer, and Sylvie Billot. 2022. A methodology for the comparison of human judgments with metrics for coreference resolution. In *Proceedings of the 2nd Workshop on Human Evaluation of NLP Systems (HumEval)*, pages 16–23, Dublin, Ireland. Association for Computational Linguistics.

Faeze Brahman, Meng Huang, Oyvind Tafjord, Chao Zhao, Mrinmaya Sachan, and Snigdha Chaturvedi. 2021. "let your characters tell their story": A dataset for character-centric narrative understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1734–1752, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ana-Isabel Duron-Tejedor, Pascal Amsili, and Thierry Poibeau. 2023. How to Evaluate Coreference in Literary Texts? *Preprint*, arXiv:2401.00238.

Abbas Ghaddar and Phillippe Langlais. 2016. WikiCoref: An English coreference-annotated corpus of Wikipedia articles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 136–142, Portorož, Slovenia. European Language Resources Association (ELRA).

Qipeng Guo, Xiangkun Hu, Yue Zhang, Xipeng Qiu, and Zheng Zhang. 2023. Dual cache for long document neural coreference resolution. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15272–15285, Toronto, Canada. Association for Computational Linguistics.

Yifan Guo, Hongying Zan, and Hongfei Xu. 2024. Dual-teacher knowledge distillation for low-frequency word translation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 5543–5552, Miami, Florida, USA. Association for Computational Linguistics.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2023. DeBERTav3: Improving deBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing. In *The Eleventh International Conference on Learning Representations*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DEBERTA: Decoding-enhanced BERT with Disentangled Attention. In *International Conference on Learning Representations*.

Lauri Karttunen. 1969. Discourse referents. In *International Conference on Computational Linguistics COLING 1969: Preprint No. 70*, Sånga Säby, Sweden.

Yuval Kirstain, Ori Ram, and Omer Levy. 2021. Coreference resolution without span representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 14–19, Online. Association for Computational Linguistics.

Vincent Labatut and Xavier Bost. 2019. Extraction and Analysis of Fictional Character Networks: A Survey. *ACM Comput. Surv.*, 52(5).

Nghia T. Le and Alan Ritter. 2024. Are Language Models Robust Coreference Resolvers? In *First Conference on Language Modeling*.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 25–32, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Kawshik S. Manikantan, Shubham Toshniwal, Makarand Tapaswi, and Vineet Gandhi. 2024. Major entity identification: A generalizable alternative to coreference resolution. In *Proceedings of The Seventh Workshop on Computational Models of Reference, Anaphora and Coreference*, pages 1–17, Miami. Association for Computational Linguistics.

Giuliano Martinelli, Edoardo Barba, and Roberto Navigli. 2024a. Maverick: Efficient and accurate coreference resolution defying recent trends. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13380–13394, Bangkok, Thailand. Association for Computational Linguistics.

Giuliano Martinelli, Francesco Molfese, Simone Tedeschi, Alberte Fernández-Castro, and Roberto Navigli. 2024b. CNER: Concept and named entity recognition. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8336–8351, Mexico City, Mexico. Association for Computational Linguistics.

Nafise Sadat Moosavi and Michael Strube. 2016. Which coreference evaluation metric do you trust? a proposal for a link-based entity aware metric. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 632–642, Berlin, Germany. Association for Computational Linguistics.

Riccardo Orlando, Pere-Lluís Huguet Cabot, Edoardo Barba, and Roberto Navigli. 2024. ReLiK: Retrieve and LinK, fast and accurate entity linking and relation extraction on an academic budget. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14114–14132, Bangkok, Thailand. Association for Computational Linguistics.

Shon Otmazgin, Arie Cattan, and Yoav Goldberg. 2022. F-coref: Fast, accurate and easy to use coreference resolution. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 48–56, Taipei, Taiwan. Association for Computational Linguistics.

Shon Otmazgin, Arie Cattan, and Yoav Goldberg. 2023. LingMess: Linguistically informed multi expert scorers for coreference resolution. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2752–2760, Dubrovnik, Croatia. Association for Computational Linguistics.

Andrew Piper, Richard Jean So, and David Bamman. 2021. Narrative theory for computational narrative understanding. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 298–311, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ian Porada and Jackie CK Cheung. 2024. Solving the challenge set without solving the task: On Winograd schemas as a test of pronominal coreference resolution. In *Proceedings of the 28th Conference on Computational Natural Language Learning*, pages 489–506, Miami, FL, USA. Association for Computational Linguistics.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using OntoNotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152, Sofia, Bulgaria. Association for Computational Linguistics.

Ina Roesiger, Sarah Schulz, and Nils Reiter. 2018. Towards coreference for literary text: Analyzing domain-specific phenomena. In *Proceedings of the Second Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 129–138, Santa Fe, New Mexico. Association for Computational Linguistics.

Kumar Shridhar, Nicholas Monath, Raghuveer Thirukovalluru, Alessandro Stolfo, Manzil Zaheer, Andrew McCallum, and Mrinmaya Sachan. 2023. Longtonotes: OntoNotes with longer coreference chains. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1428–1442, Dubrovnik, Croatia. Association for Computational Linguistics.

Shubham Toshniwal, Sam Wiseman, Allyson Ettinger, Karen Livescu, and Kevin Gimpel. 2020. Learning to Ignore: Long Document Coreference with Bounded Memory Neural Networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8519–8526, Online. Association for Computational Linguistics.

Shubham Toshniwal, Patrick Xia, Sam Wiseman, Karen Livescu, and Kevin Gimpel. 2021. On generalization in coreference resolution. In *Proceedings of the Fourth Workshop on Computational Models of Reference, Anaphora and Coreference*, pages 111–120, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hardik Vala, David Jurgens, Andrew Piper, and Derek Ruths. 2015. Mr. bennet, his coachman, and the archbishop walk into a bar but only one of them gets recognized: On the difficulty of detecting characters in literary texts. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 769–774, Lisbon, Portugal. Association for Computational Linguistics.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Sixth Message Understanding Conference (MUC-6): Proceedings of a Conference Held in Columbia, Maryland, November 6-8, 1995*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. 2024. Qwen2 technical report. *Preprint*, arXiv:2407.10671.

Wenzheng Zhang, Sam Wiseman, and Karl Stratos. 2023. Seq2seq is all you need for coreference resolution. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 11493–11504, Singapore. Association for Computational Linguistics.

A Resource Details

The authors of LiSCU (Brahman et al., 2021) do not directly release their data but instead redirect to a GitHub repository to scrape the Internet Archive and obtain the full dataset.[12] We fix broken links, remove some restrictive filters from their codebase that skip minor characters, and run their scripts to save a local copy of LiSCU in a local DB. We then export a list of 24,985 tuples, each specifying a character name, the author and title of the book it is linked to, and the internet study guide that reports this information. The total number of books extracted from LiSCU is 1,707, each linked to 14.6 characters on average.

For Wikidata, we derive our own SPARQL query to use three main Wikidata relations: *characters* (P674), which links an item such as a book or a film to characters that appear in it; *author* (P50), which links to the author of a specific work; and *title* (P1476), the published name of a literary work or newspaper article or others. Running this query results in 17,324 distinct (author, title) tuples, each linked on average to 1.54 characters. This highlights some issues of the Wikidata Knowledge Graph, as the character relation has poor coverage and there is no reliable way to restrict an item identified by an (author, title) tuple to be a work of literary text.

[12] https://github.com/huangmeng123/lit_char_data_wayback

We then attempt to match each (author, title) obtained from the concatenation of LiSCU and Wikidata to a valid book in Project Gutenberg. We use the `gutenbergpy` library[13], which includes functions to search through the whole Project Gutenberg book metadata for an exact token match between our (author, title) tuples and the author and title information in Gutenberg. There is a possibility of multiple (author, title) matches in our character tuples, as there might be some overlap between the sources they are derived from (Wikidata and the three different study guides contained in LiSCU). We simply select the source which contains the largest number of characters for a given book. We manually validate the links between the list of characters of a book and its raw text extracted from Gutenberg, as well as manually cleaning each Gutenberg book to remove headings, placeholders, notes, etc. This leaves us with our base resource: a set of 52 cleaned books obtained from Project Gutenberg, each linked to a list of characters that appear in the book.

We also include in our corpus *Animal Farm* by George Orwell, as contributed by Guo et al. (2024), resulting in a total number of 53 books.

Licenses Books from Project Gutenberg are available in the public domain in the United States, and the authors of LiSCU (Brahman et al., 2021) make their code freely accessible under an MIT license. The Animal Farm annotation provided by Guo et al. (2024) is available under the permissive BSD 3-clause license. We are also allowed to make full use of LitBank under the Creative Commons Attribution 4.0 International License. Qwen2 7B is licensed with Apache 2.0. Both ReLiK (Orlando et al., 2024) and Maverick (Martinelli et al., 2024a) are licensed with the Attribution-NonCommercial-ShareAlike 4.0 International License and therefore can be used for research purposes.

B Entity Linking on LitBank

Our main aim is to create a dataset that can be used to fine-tune ReLiK (Orlando et al., 2024) to perform Character Linking on the narrative text. To accomplish this, we adapt the LitBank Coreference Resolution dataset based on the following intuition: if we are able to assign a character name to each co-referring cluster of mentions associated to a character, we can then re-frame the task as linking each specific mention to its character name. More practically, the coreference data in LitBank comes with a categorization of entities across six ACE 2005[14] categories (people, facilities, locations, geopolitical entities, organizations and vehicles); we therefore only keep coreference clusters where all mentions are of type *people*, including both named entities and common nouns, which are crucial for many downstream applications (Martinelli et al., 2024b). Moreover, for each mention in a cluster, we are given its manually-validated part-of-speech tag. We filter out any pronoun mentions from all the clusters, as they are not usually tagged in Entity Linking settings. We then manually name each remaining cluster, taking inspiration from the most frequently occurring proper nouns in a cluster but also taking clues from the book (e.g., naming a cluster "Tom Sawyard" even if the most frequently occurring PROP is "Tom").

This leaves us with a complete Character Linking dataset, i.e., an Entity Linking dataset where all the entities are anthropomorphized characters. The index of possible entities is the set of named coreference clusters of a book and is different for each book.

We then follow the instructions in the ReLiK repository[15] to fine-tune the ReLiK Reader on the LitBank training set. Evaluating on the held-out testing set returned an F1 score of 83.1.

Prompt
I will give you an excerpt from a book with a highlighted mention of a character with []. You will need to answer if the assigned character is correct (Yes), or not (No). Book excerpt: $book Does the mention [$mention] correspond to the character $character? (Yes/No)

Table 5: Prompt template for the Cluster Refinement step. The dollar sign ($) indicates a template variable.

C Prompting LLMs for Cluster Refinement

We detail our prompt template in Table 5. We apply it with a specific mention from a character cluster, together with the 400 words that surround the mention as local context. We do not provide any in-context examples, as we found the model to

[13] https://github.com/raduangelescu/gutenbergpy

[14] https://catalog.ldc.upenn.edu/LDC2006T06

[15] https://github.com/SapienzaNLP/relik

perform reliably in a zero-shot setting (line "LLM filtering" in Table 2).

A possible edge-case of this setting might be when there is not enough context for the LLM to accurately determine if the mention is correctly linked to a character. We performed a small experiment by prompting the LLM with an additional clause (i.e., "If not enough context is provided answer No") and we measured a negligible difference compared to the values presented in Table 2: with the new prompt, the model performances do not change in precision (+0.0), and decrease in recall and F1 score (-0.2 and -0.1 respectively). We believe that the equivalence of the two prompting strategies is due to: i) the simplicity of the task, as the considered mentions are explicit (no pronominal mentions are detected by Character Linking) and 42.7% of them exactly match the name of the character (Pattern Matching baseline in Table 2); and ii) the LLM being able to answer correctly without solely relying on context (i.e., answering correctly because of its parametric memory).

D Manual-annotation Details

As in the annotation of *Animal Farm* (Guo et al., 2024), we provide the annotators with a list of characters appearing in each book, and they are tasked with extracting co-referring mentions of each character. We follow previous works (LitBank, OntoNotes) in selecting as co-referring mentions spans of text that are part of one of the following categories: proper nouns (for example, *Siddhartha*); common nouns (*the gentleman*); personal pronouns (*he*, *she*); possessive pronouns (*his*, *hers*).

We also decide to mark the maximal extent of a span, resulting in noun phrases such as *"one of the most eminent physicians"*, *"a man whom nobody cared anything about"* or *"the Right Honourable Lady Catherine de Bourgh"*. This latter example showcases our approach to honorifics, which we include in the maximal span without annotating them separately.

Regarding singleton mentions (i.e., noun phrases that do not co-refer with other mentions), we allow them to be tagged by our annotators, but because the task pertains to identifying co-referring mentions of a list of characters for each book, there are no instances of singleton mentions in BOOKCOREF$_{gold}$.

Finally, we also instruct annotators to avoid annotating mentions that can be linked to more than one antecedent mention (referred to as *split-antecedent*), in line with most current Coreference Resolution datasets.

E Current Systems Limitations

In this section, we detail the limitations of current systems when applied to this new book-scale setting.

Discriminative models Discriminative models formulate the CR task as a classification problem. They usually adopt the same two-step formulation, in which the model learns to first extract the mentions and then predicts their coreference relations. Recent models use an underlying encoder-only pre-trained Transformer architecture to encode the input documents. This give rise to two main limitations: i) pre-trained Transformers usually have a fixed maximum input length and, ii) their attention mechanism involves a quadratic computational complexity with respect to the input text. These two characteristics strongly limit their applicability to the proposed book-scale setting, in which input books can reach more than 600.000 tokens.

Sequence-to-Sequence models Sequence-to-sequence models have recently proven particularly robust for resolving coreference relations in well-established CR benchmarks(Bohnet et al., 2023; Zhang et al., 2023). However, the limitations that make the usage of discriminative models difficult in the full-book setting are even more marked when considering Sequence-to-Sequence architectures. In fact, these models usually rely on very large pre-trained Transformers with several billions of parameters and many works report that their applicability is limited by their computational requirements (Zhang et al., 2023; Martinelli et al., 2024a). An additional limitation is inherently implied by their application to the coreference task, which requires re-generating the full input text with the addition of special tokens that denote coreference. In this regard, Bohnet et al. (2023) reports problems of hallucinations and ambiguous matches of mentions to the input when dealing with the documents in OntoNotes. For these reasons, we note that the proposed book-scale setting is particularly challenging for generative systems with current formulations, and more advancements are needed to adapt them to this extended-length scenario.

LLMs The limitations we raise for large generative Coreference Resolution systems are exacerbated when attempting to apply Large Language Models (LLMs) to this task. To the best of our knowledge, there is no clear consensus on how LLMs should be used to extract and resolve mentions of co-referring entities without the pre-identification of mentions or the appearance of hallucinated text. Previous work also shows that, on the task of Coreference Linking, LLMs do not surpass the performance of smaller, bespoke models on medium-scale settings (Le and Ritter, 2024; Porada and Cheung, 2024). Moreover, the scale of our benchmark would constrain us to use only LLMs that have a context window larger than twice the length of the longest document in BOOKCOREF$_{gold}$, i.e., more than 300,000 tokens. We are excited to see how our benchmark could contribute to convincing progress on these challenges.

F Comparison Systems Details

In this section, we dive into further details regarding the experimental setup presented in Section 4.2.

BookNLP The BookNLP pipeline[16] is a collection of modules based on BERT embeddings, each tasked with carrying out a specific task, such as Entity Recognition, Quotation Attribution, Character Clustering and Coreference Resolution. The performance of this model on Literary Coreference Resolution tasks and its widespread adoption among research in the Digital Humanities field make it an interesting baseline. Unfortunately, the library does not provide any option to fine-tune the model for CR, which we speculate might be because of the complex interplay of pipeline components that it relies upon to resolve co-referring mentions at the book scale. Nevertheless, we adopt it as a baseline in our off-the-shelf setting on both BOOKCOREF$_{gold}$ and SPLIT-BOOKCOREF$_{gold}$.

Longdoc Introduced by Toshniwal et al. (2020, 2021), it uses an incremental method for mention clustering, in which mentions are scored against a learned vector representation of previously built clusters. Specifically, Longdoc initializes a "memory architecture" that saves previous clusters to compare against. There are two policies that the memory architecture adopts: "unbounded", where all clusters are kept for mention clustering; or "least-recently used", where once the maximum number of clusters is reached, the cluster that was used least recently is no longer considered for mention clustering and is thus evicted from memory. In our experiments, we keep with the default choice encoded in Longdoc and adopt the unbounded policy.

We were able to train Longdoc from scratch on our BOOKCOREF$_{silver}$ by using its codebase.[17] Our interventions are minimal: we adapt our dataset to their format and run the training, leaving all hyperparameters unchanged. Notably, Longdoc uses Longformer as an encoder model, from which the maximum input length is limited to 4096 tokens. At inference time, longdoc encodes documents in sliding windows and performs the clustering step iteratively over the full input.

Dual cache Guo et al. (2024) developed this model as a modification of the Longdoc memory architecture. They develop a mechanism that maintains two "caches", one controlled by a least-recently used policy (L-cache, or local) and the other by a least-frequently used policy (G-cache, or global). The model has control over the location of clusters and can swap them between L- and G-cache or remove them entirely depending on the policy value.

The experiments reported by Guo et al. (2024) are based on fine-tuning an existing Longdoc model, but they do not release any model weights. We therefore replicate their setup by initializing the Dual cache encoder from the available Longdoc encoder and fine-tuning the whole model on LitBank, adopting the hyperparameters specified in their repository.[18] Having chosen the configuration with sizes of the L- and G-cache set to 25 each, we obtain comparable results: 79.68 CoNLL F1 on LitBank compared to their reported 78.8, and 36.8 CoNLL F1 on Animal Farm against their 36.3.

We then fine-tune this LitBank-trained version of Dual cache on BOOKCOREF$_{silver}$, without changing any hyper-parameter.

Notably, Dual cache is also based on Longformer.

Maverick Maverick (Martinelli et al., 2024a) is an encoder-only model that resolves coreference in a single pass. We test the Maverick$_{xl}$ architecture, presented in Section 4.2, which is based on the

[16] https://github.com/booknlp/booknlp

[17] https://github.com/shtoshni/fast-coref

[18] https://github.com/QipengGuo/dual-cache-coref/tree/main

weights of MAVERICK-MES-LITBANK.[19] Specifically, this model is based on DeBERTa-v3-large and pre-trained on LitBank, therefore its maximum input length is around 25,000 tokens.

G Coreference Metrics on Long-document BOOKCOREF

In Section 5, we show comparison systems reaching higher performances when performing CR on small texts compared to full books. This finding raises an interesting research question: do models under-perform when tested on full books or are metrics penalizing the same predictions when evaluating on the book-scale setting? In this Section, we provide further experiments to analyze this phenomenon.

G.1 Experimental Setting

We benchmark both off-the-shelf and trained systems on the two settings presented in Section 4.1 and propose a third setting to obtain further insights, BOOKCOREF$_{gold+window}$. We now detail the three proposed settings:

- BOOKCOREF$_{gold}$, the full-book setting. Models take in input full-books and predict book-level coreference clusters that are scored against book-level gold clusters.
- SPLIT-BOOKCOREF$_{gold}$, the split-book setting: Models take in input books split into independent windows of 1500 tokens and produce window-level clusters, which are scored against the respective window-level gold clusters.
- BOOKCOREF$_{gold+window}$, an intermediate setting, in which full-book predictions are evaluated in windows. Specifically, models take in input full books and predict book-level coreference clusters, as in the full-book setting. Then, performance is computed by considering only the predictions that refer to the specific windows used in SPLIT-BOOKCOREF$_{gold}$, scoring them against the respective window-level gold clusters.

Introducing our third intermediate setting is crucial for our investigation in the behavior of coreference metrics and models in our book-scale scenario.

[19] https://huggingface.co/sapienzanlp/maverick-mes-litbank

Comparing the results between BOOKCOREF$_{gold}$ and BOOKCOREF$_{gold+window}$ is useful to evaluate whether the same predictions are underestimated by traditional coreference metrics. On the other hand, comparing the outcome of SPLIT-BOOKCOREF$_{gold}$ and BOOKCOREF$_{gold+window}$ is useful to investigate changes in performances when having the full book context.

G.2 Results

Performance of comparison systems In Table 6 we report the results of our comparison systems in the three proposed settings. Interestingly, results on BOOKCOREF$_{gold+window}$ are consistently higher compared to the ones measured in BOOKCOREF$_{gold}$, indicating that traditional coreference metrics are biased towards lower performances when evaluating full-book predictions. Furthermore, when evaluated on BOOKCOREF$_{gold+window}$, the score increase is particularly high for Dual-cache, suggesting that this model is locally accurate but lacks full-book consistency. The same cannot be said for Longdoc, which in the BOOKCOREF$_{gold+window}$ setting confirms the high performances obtained on full-book predictions. In general, all the systems perform better when taking in input medium-sized texts, as on SPLIT-BOOKCOREF$_{gold}$, suggesting that the current models are particularly tailored for processing shorter lengths.

Coreference metrics In the full book setting, we notice a low agreement between the MUC and CEAF$_{\phi_4}$ metrics. In particular, MUC scores are typically very high, whilst CEAF$_{\phi_4}$ instead assigns a low evaluation to the same predictions. This gap between these two metrics is less evident on BOOKCOREF$_{gold+window}$, in which the same predictions are evaluated in smaller windows. Since analyzing this phenomenon is not in the scope of this paper, we leave this research direction to future work. Nevertheless, we believe that a more detailed study is needed for analyzing metrics behavior in full-book settings, and more attention should be devoted to the interpretation of their scores.

	BOOKCOREF$_{gold}$				BOOKCOREF$_{gold+window}$				SPLIT-BOOKCOREF$_{gold}$			
	MUC	B^3	CEAF$_{\phi_4}$	CoNLL	MUC	B^3	CEAF$_{\phi_4}$	CoNLL	MUC	B^3	CEAF$_{\phi_4}$	CoNLL
Off-the-shelf												
BookNLP	83.1	40.9	2.4	42.2	81.3	54.3	23.4	53.0	81.1	52.3	18.7	50.6
Longdoc	79.9	52.1	7.9	46.6	80.3	64.2	34.2	59.4	80.7	63.8	39.0	61.2
Dual cache	82.3	41.0	4.2	42.5	82.6	67.4	41.3	63.7	82.9	67.8	43.9	64.8
Maverick$_{xl}$	81.2	35.6	6.1	41.2	80.9	55.9	26.2	54.3	83.8	69.5	46.1	66.5
Fine-tuned on BOOKCOREF$_{silver}$												
Longdoc	93.5	**62.4**	**45.3**	**67.0**	80.8	**75.3**	**62.2**	**76.2**	91.2	74.9	65.7	77.1
Dual cache	92.5	48.0	16.9	52.5	90.7	73.5	61.4	75.2	91.2	74.5	66.2	77.3
Maverick$_{xl}$	**94.3**	55.3	33.4	61.0	**91.7**	72.8	55.6	73.4	**92.7**	**82.2**	**71.9**	**82.2**

Table 6: Comparison between off-the-shelf models and systems trained on BOOKCOREF$_{silver}$, tested on BOOKCOREF$_{gold}$ SPLIT-BOOKCOREF$_{gold}$ and BOOKCOREF$_{gold+window}$. For each system, we report F1 measures of MUC, B^3 and CEAF$_{\phi_4}$, and use their average, CoNLL-F1, as the main evaluation criteria. We highlight in **bold** the best measures of trained models, and underline best off-the-shelf results.

Stage	Annotation
Character Linking (CL)	[Miss Bingley]$_{\text{Caroline Bingley}}$ sees that [her brother]$_{\text{Mr. Darcy}}$ is in love with you and wants him to marry [Miss Darcy]$_{\text{Georgiana Darcy}}$.
CL + LLM Filtering	[Miss Bingley]$_{\text{Caroline Bingley}}$ sees that her brother is in love with you and wants him to marry [Miss Darcy]$_{\text{Georgiana Darcy}}$.
CL + Window Coreference	[Miss Bingley]$_{\text{Caroline Bingley}}$ sees that [[her]$_{\text{Caroline Bingley}}$ brother]$_{\text{Mr. Darcy}}$ is in love with [you]$_{\text{Jane Bennet}}$ and wants [him]$_{\text{Mr. Darcy}}$ to marry [Miss Darcy]$_{\text{Georgiana Darcy}}$.
CL + LLM Filtering + Window Coreference	[Miss Bingley]$_{\text{Caroline Bingley}}$ sees that [[her]$_{\text{Caroline Bingley}}$ brother]$_{\text{Charles Bingley}}$ is in love with [you]$_{\text{Jane Bennet}}$ and wants [him]$_{\text{Charles Bingley}}$ to marry [Miss Darcy]$_{\text{Georgiana Darcy}}$.
CL	When , after examining [the mother]$_{\text{Lady Catherine de Bourgh}}$, in whose countenance and deportment she soon found some resemblance of [Mr. Darcy]$_{\text{Mr. Darcy}}$, she turned her eyes on [the daughter]$_{\text{Charlotte Lucas}}$, she could almost have joined in [Maria]$_{\text{Maria Lucas}}$'s astonishment at her being so thin and so small .
CL + LLM Filtering	When , after examining the mother , in whose countenance and deportment she soon found some resemblance of Mr. Darcy , she turned her eyes on the daughter , she could almost have joined in [Maria]$_{\text{Maria Lucas}}$'s astonishment at her being so thin and so small .
CL + Window Coreference	When , after examining [the mother]$_{\text{Lady Catherine de Bourgh}}$, in whose countenance and deportment [she]$_{\text{Elizabeth Bennet}}$ soon found some resemblance of [Mr. Darcy]$_{\text{Mr. Darcy}}$, [she]$_{\text{Elizabeth Bennet}}$ turned [her]$_{\text{Elizabeth Bennet}}$ eyes on [the daughter]$_{\text{Charlotte Lucas}}$, [she]$_{\text{Elizabeth Bennet}}$ could almost have joined in [Maria]$_{\text{Maria Lucas}}$'s astonishment at [her]$_{\text{Charlotte Lucas}}$ being so thin and so small .
CL + LLM Filtering + Window Coreference	When , after examining [the mother]$_{\text{Lady Catherine de Bourgh}}$, in whose countenance and deportment [she]$_{\text{Elizabeth Bennet}}$ soon found some resemblance of [Mr. Darcy]$_{\text{Mr. Darcy}}$, [she]$_{\text{Elizabeth Bennet}}$ turned [her]$_{\text{Elizabeth Bennet}}$ eyes on [the daughter]$_{\text{Miss de Bourgh}}$, [she]$_{\text{Elizabeth Bennet}}$ could almost have joined in [Maria]$_{\text{Maria Lucas}}$'s astonishment at [her]$_{\text{Miss de Bourgh}}$ being so thin and so small .

Table 7: Error analysis of our BookCoref Pipeline predictions on *Pride and Prejudice* from our test set. The table examines the three main pipeline steps as defined in Table 2 (Cluster Initialization, Cluster Refinement, and Cluster Expansion), through two examples. The first example demonstrates effective LLM Filtering, which correctly removes the mention "[her brother]$_{\text{Mr. Darcy}}$" that should link to 'Charles Bingley'. Without this filtering, Cluster Expansion would propagate the error by including nearby pronouns. When applied after LLM Filtering, Cluster Expansion successfully identifies the correct character by linking coreferential mentions across a wider context. The second example illustrates the pipeline's robustness even when LLM Filtering over-corrects by removing valid mentions like "[Mr. Darcy]$_{\text{Mr. Darcy}}$". The Cluster Expansion step recovers these filtered mentions while also correcting incorrectly linked mentions from the initial clustering ("[the daughter]$_{\text{Charlotte Lucas}}$").

	Book name	Gutenberg key	Number of tokens	Number of characters
	The Pilgrim's Progress	7088	30610	45
	The Boxcar Children	42796	30819	11
	Dr. Jekyll and Mr. Hyde	42	31127	8
G	Animal Farm*	-	36504	20
G	Siddhartha	2500	47785	9
	The Awakening	160	60031	26
	Pudd'nhead Wilson	102	63061	20
	O Pioneers!	24	67463	12
	The Hound of the Baskervilles	2852	70582	19
	The Prince and the Pauper	1837	81947	21
	Silas Marner	550	87218	33
	Northanger Abbey	121	92861	17
	My Ántonia	19810	95698	38
	Kidnapped	421	98379	21
	Persuasion	105	99102	20
	Fathers and Sons	47935	99902	27
	The Tale of Genji	66057	106868	20
	Pan Tadeusz	28240	136711	10
	Sense and Sensibility	161	142607	18
	The Mayor of Casterbridge	143	143129	13
G	Pride and Prejudice	1342	146495	38
	The Talisman	1377	155678	10
	Moll Flanders	370	159287	14
	In Search of the Castaways	46597	166845	10
	A Tale of Two Cities	98	170076	14
	The Last of the Mohicans	27681	172400	12
	The Return of the Native	122	174304	18
	The Jungle	140	177580	46
	The Way of All Flesh	2084	185562	39
	Emma	158	190921	24
	The Canterbury Tales	2383	205165	45
	Main Street	543	209394	84
	The Red and the Black	44747	222200	24
	The Man in the Iron Mask	2759	222328	33
	Ivanhoe	82	224176	70
	North and South	4276	224604	11
	Little Women	514	229337	61
	Jane Eyre	1260	231978	34
	Uncle Tom's Cabin	203	233220	25
	The Deerslayer	3285	253815	15
	Pamela	6124	267364	35
	The Three Musketeers	1257	292259	31
	The Woman in White	583	294178	13
	Twenty Years After	1259	315692	20
	Middlemarch	145	378961	35
	The Pickwick Papers	580	388962	88
	Our Mutual Friend	883	417160	17
	Little Dorrit	963	418851	15
	The Brothers Karamazov	28054	435055	12
	Bleak House	1023	437143	66
	Don Quixote	996	473968	31
	War and Peace	2600	675324	32
	Les Miserables	135	689400	45

Table 8: List of all books that compose BOOKCOREF, together with their Gutenberg ID, number of tokens and number of characters. With the color green and the letter G we identify books that were manually annotated to create BOOKCOREF$_{gold}$. *: *Animal Farm* is included from Guo et al. (2024) with no further changes.