Cross-Genre Authorship Attribution via LLM-Based Retrieve-and-Rerank

Anonymous ACL submission

Abstract

Authorship attribution (AA) is the task of identifying the most likely author of a query document from a predefined set of candidate authors. We introduce a two-stage retrieve-and-rerank framework that finetunes LLMs for cross-genre AA. Unlike the field of information retrieval (IR), where retrieve-and-rerank is a de facto strategy, cross-genre AA systems must avoid relying on topical cues and instead learn to identify author-specific linguistic patterns that are independent of the text's subject matter (genre/domain/topic). Consequently, for the reranker, we demonstrate that training strategies commonly used in IR are fundamentally misaligned with cross-genre AA, leading to suboptimal behavior. To address this, we introduce a targeted data curation strategy that enables the reranker to effectively learn author-discriminative signals. Using our LLM-based retrieve-and-rerank pipeline, we achieve substantial gains of 22.3 and 34.4 absolute Success@8 points over the previous state-of-the-art on HIATUS's challenging HRS1 and HRS2 cross-genre AA benchmarks.

1 Introduction

The field of authorship analysis has a long history, dating back at least as far as 1440 when Lorenzo Valla used textual analysis to prove that the *Donation of Constantine* was a forgery (Valla, 1922). It has played a central role in determining the authors of significant works such as the *The Federalist Papers* (Mosteller and Wallace, 1984), as well as contributing to the resolution of criminal cases like the *Unabomber* bombing investigation (Leonard et al., 2017).

Due to its long-standing history and continued research interest, authorship analysis has been explored across a wide range of settings. These include variations in task formulation (authorship attribution vs. verification) Tyo et al. (2022), text length (short Layton et al. (2010) vs. long Nguyen et al. (2023)), languages Israeli et al. (2025), domain mismatch (e.g., differences in topic or genre between training and test data) Wegmann et al. (2022), author set assumptions (closed-set Barlas and Stamatatos (2020) vs. open-set Badirli et al. (2021) scenarios), size of the test corpus (Rivera-Soto et al., 2021), etc. Given these nuanced distinctions, it is important to clearly define the scope of this work.

We address the task of authorship attribution and target it as a *ranking task*: given a query document with unknown authorship, the objective is to rank candidate documents by their likelihood of sharing the same author. All documents in the test corpora are in English. Our approach generalizes across multiple datasets, spanning long documents (median of $\sim$ 600 words per document) and medium-length documents (median of $\sim$ 200 words per document). The setting is cross-genre, where the query and the relevant candidate document differ not only in topic but also in genre - for example, a news article in politics as the query and a StackExchange post about literature as the candidate (see Table 1); the system must generalize to authors unseen during training and therefore cannot rely on author-specific classifiers; at least one candidate document is guaranteed to have the same author as the query; and the candidate author pool is large, comprising tens of thousands of authors.

Building a ranking system that scales to a large candidate pool requires both efficiency and accuracy. To achieve this, the field of IR commonly adopts a two-stage approach (Gao et al., 2021a; Ma et al., 2024): an efficient retrieval stage followed by a more accurate but compute-intensive reranking stage. In this work, we extend this paradigm to AA, focusing specifically on the cross-genre setting. The first stage involves retrieval via a bi-encoder where each document is independently encoded into a vector representation, and the likelihood that two documents share the same author is quantified

using the dot product of their vectors. In the second stage, we use a cross-encoder to rerank the retrieved documents. The reranker takes both the query and a candidate document as input to directly compute a relevance score. Due to its joint modeling of the query-candidate pair, the reranker achieves higher accuracy than the retriever, but at the cost of greater computational expense.

Our main contributions are as follows:

- Integrating LLMs: Inspired by the effectiveness of LLMs in IR (Ma et al., 2024), we leverage them for both the retriever and the reranker. While prior work has primarily focused on leveraging LLMs' zero/few-shot prompting (Hung et al., 2023; Huang et al., 2024), to the best of our knowledge, ours is the first to explore fine-tuning LLMs specifically for AA.

- Effective reranker for AA: While prior work has framed AA as a ranking task (Rivera-Soto et al., 2021; Fincke and Boschee, 2024), existing approaches have relied primarily on a retrieval stage alone. In this work, we advance this line of research by introducing a reranking component. We show that building an effective reranker for cross-genre AA is non-trivial and we address this challenge through targeted training data curation.

We name our system SADIRI-V2[1].

2 Methodology

We formalize the AA task in Section 2.1. We describe our retriever and the reranker in Sections 2.2 and 2.3 respectively.

2.1 Task definition

The objective of an AA system is to give a likelihood score for two documents to have the same author (Rivera-Soto et al., 2021). Concretely, we have a set of query documents $\mathcal{Q} = \{q_1, q_2, \cdots, q_{|Q|}\}$, where q_i is the ith query document and $|Q|$ is the total number of documents in the query set. Similar to the query set, we also have a set of candidate documents, $\mathcal{C}$. For each document entry in the query collection, the AA system needs to rank the candidate set such that candidate documents which are likely to have the same author as the query document are ranked higher (Rivera-Soto et al., 2021). For each query document, q_i, we assume that there is at least one candidate document, c_j, called the needle, which has the same author as the query. Documents in the candidate set that are not written by the author of q_i act as haystack (or distractors).

We focus on the setting where the query and needle documents are guaranteed to differ both in genre and topic. This challenge is compounded by the presence of haystack documents that are semantically similar to both the query and the needle (see Table 1). To succeed, an effective AA system must ignore topical cues which could lead to matching with unrelated but topically similar documents while accurately capturing authorial features that link the query to the needle (Wegmann et al., 2022).

2.2 Retriever

Architecture: The first stage of our two-stage pipeline prioritizes efficiency and uses a standard bi-encoder architecture for retrieval. Unlike in prior works on AA, we fine-tune LLMs to serve as the retriever. Given a document d, the transformer LLM outputs a sequence of hidden states $(h_d^1, h_d^2, \ldots, h_d^L)$, where h_d^i is the final-layer hidden representation of the ith token, and L is the tokenized document length. We apply mean pooling over these token representations to obtain a fixed-length vector: $v_{\text{agg}}(d) = \text{mean}(h_d^1, h_d^2, \ldots, h_d^L)$. This vector lies in R^E, where E corresponds to the hidden dimension of the transformer. A learnable linear projection is then applied to produce the final document embedding:

$$v(d) = W v_{\text{agg}}(d) + b, \quad (1)$$

where $W \in R^{D \times E}$ is a projection matrix and $b \in R^D$ is a bias vector. In our experiments, we set $D = E/2$.

For training, we construct each batch with N distinct authors, including exactly two documents per author, resulting in a total of $2N$ documents per batch. The bi-encoder is trained using supervised contrastive loss (Khosla et al., 2020), $l = \frac{1}{2N} \sum_{q=1}^{2N} l_q$ where:

$$l_q = -\log \frac{\exp\left(s(d_q, d_q^+)/\tau\right)}{\sum_{d_c \in \{d_q^+\} \cup D^-} \exp\left(s(d_q, d_c)/\tau\right)} \quad (2)$$

In the above equation, $s(d_q, d_c)$ denotes the score for a query and a candidate document to share the

[1]The original SADIRI (Fincke and Boschee, 2024) was a RoBERTa (Liu et al., 2019) based retriever only model and is one of our baselines against which we compare SADIRI-V2.

same author, d_q^+ is the (positive) document in the batch that is written by the same author as d_q, D^- denotes the set of negative documents, i.e. documents not authored by the author of d_q, and τ is a positive scalar (temperature) hyperparameter. For the bi-encoder, we calculate the score by taking the dot product between the two document vectors, i.e. $s(d_q, d_c) = v(d_q) \cdot v(d_c)$.

Training batches with hard negatives: For a document d_q, a negative document $d^- \in D^-$ is considered hard if $v(d_q) \cdot v(d^-)$ is large. Hard negatives lead to a larger denominator in contrastive loss and thus contribute to faster model convergence. The importance of hard negative documents is extensively studied in the representation learning literature such as in IR (Karpukhin et al., 2020). We adopt SADIRI 's (Fincke and Boschee, 2024) strategy for constructing training batches using in-batch negative sampling, where D^- includes all $2N - 2$ documents in the batch which are not authored by the writer of d_q. The training documents are first clustered using k-means based on the cosine distance between their vectors, and each cluster is assigned to a batch. The underlying intuition is that documents grouped together by clustering, when not sharing the same author, tend to be hard negatives for each other. See Appendix C for details on how the batches are constructed via clustering.

2.3 Reranker

Architecture: We use cross-encoders as our reranker. Given a query and a candidate document, the model jointly encodes the pair to produce a similarity score reflecting the likelihood of shared authorship. This joint encoding enables the reranker to have higher discriminative power than the retriever.

Because the reranker jointly encodes the query and the candidate documents, its run-time complexity is $|\mathcal{Q}| \times |\mathcal{C}|$, where $|\mathcal{Q}|$ and $|\mathcal{C}|$ are the number of documents in the query and the candidate collections respectively [2]. Because of the reranker's quadratic run-time complexity, it is infeasible to rank all candidate documents for each query input, and thus, the reranker is applied only to the top-k candidates retrieved in the first stage.

We fine-tune LLMs as cross-encoders by concatenating the query and candidate documents using a special delimiter. The reranker computes an

[2] On the other hand, the first-stage retriever's run-time complexity is linear in the number of documents in the collection: $|\mathcal{Q}| + |\mathcal{C}|$.

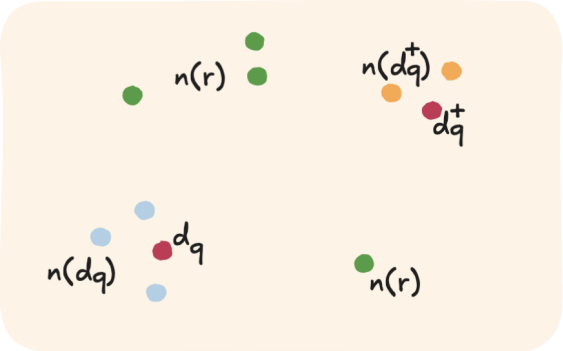

Figure 1: Negative documents, i.e. documents not written by the author of the query document, can be categorized into three distinct groups: $n(d_q)$ are documents close to the query document, $n(d_q^+)$ are close to the positive document and $n(r)$ are random documents with no particular bearing to either d_q or d_q^+.

authorship similarity score as:

$$s(d_q, d_c) = W h^L_{(d_q, d_c)}, \quad (3)$$

where $h^L_{(d_q,d_c)} \in R^E$ is the final-layer hidden state of the last input token, E is the LLM's hidden dimension, and $W \in R^{1 \times E}$ is a learned projection matrix. The reranker is trained using the contrastive loss in Eq. 2, with similarity scores computed as per Eq. 3.

Hard negatives to train the reranker: We construct each training batch using N authors, with two documents per author: one serving as the query document (d_q) and the other as the positive candidate (d_q^+). For each query, m negative documents not authored by the query author are selected to form D^-. Thus, in total, each d_q is compared against $(1 + m)$ candidate documents.

Based on its similarity to the query and the positive document, each negative document falls into one of three categories: $n(d_q)$ - close to the query document, $n(d^+)$ - close to the positive document, or $n(r)$ - random.

Note that this categorization of negatives is fundamentally different from the choice of negatives available in IR. In IR, the negatives for the reranker are typically drawn from the retriever's top-ranked candidates (Gao et al., 2021a), i.e. those with a high $s(d_q, d_c)$ score in Eq. 2. In addition, IR retrievers typically do not explicitly model the closeness of candidate documents to the positive, i.e. $s(d_q^+, d_c)$ is not available. This distinction arises from a fundamental asymmetry in IR: queries are often short and structurally different from candidate documents. In contrast, AA involves com-

paring two documents, making the query and the candidate symmetric in nature.

We demonstrate that the reranker's performance is strongly influenced by the distribution of negative samples used for training. Sampling exclusively from $n(d_q)$ (analogous to IR-style retrieval) yields insufficient contrastive signal for learning robust cross-genre authorship representations. We also demonstrate that sampling negatives from all three categories, $n(d_q) + n(d_q^+) + n(r)$, leads to the best cross-genre system performance.

3 Experiments

In this section, we evaluate our retrieve-and-rerank approach on the cross-genre authorship attribution task. Section 3.1 describes our training data selection strategy for effective cross-genre AA, section 3.2 describes HIATUS's cross-genre test corpus, section 3.3 outlines the evaluation metrics, section 3.4 presents the baseline methods, and section 3.5 provides implementation details.

3.1 Curating training data for cross-genre AA

Building a strong cross-genre AA model requires guiding it to rely on authorial style rather than content. This is typically achieved through two strategies: (1) carefully selecting training data to discourage content-based matching (Wegmann et al., 2022; Fincke and Boschee, 2024), and (2) introducing inductive bias by masking content-bearing words, such as replacing them with their corresponding POS tags (Wang et al., 2023).

We adopt the first strategy. Starting with web-scraped documents across 8 diverse genres, we apply the following filtering steps: remove documents with fewer than 350 words[3]; discard authors with fewer than 2 or more than 50 documents[4]. For each author, we calculate the SBERT-based[5] (Reimers and Gurevych, 2020) cosine similarity between all document pairs and add the most dissimilar document pair to our training collection if the cosine similarity is less than a predefined threshold. To ensure a fair comparison, we select SADIRI 's threshold of 0.2. SBERT based filtering ensures that d_q and d_q^+ are topically dissimilar; hence the bi-encoder/reranker models must learn features other than topic when distinguishing the positive from the negative documents. The final training set contains 395,118 documents from 197,559 authors, with exactly two documents per author. See Appendix A for additional details.

3.2 Test Dataset

HIATUS[6] is an IARPA[7] initiative focused on developing human-usable systems for attribution of authorship and protection of author privacy. We test the robustness of our approach on HIATUS's HRS1 and HRS2 English test sets[8], which contain long and medium-length documents, respectively. HRS1 includes only documents with at least 350 words, with a median length of 581 words. In contrast, HRS2 consists of medium-length documents, each with at least 80 words and a median length of 206 words. Both HRS1 and HRS2 include query and candidate documents that span five distinct genres. We provide the distribution of each genre in these datasets in Appendix B. For each genre, two types of document are provided: background documents, scraped from the Web, and foreground documents, written specifically for the HIATUS program. The foreground authors contribute documents in multiple genres, enabling a controlled and diverse evaluation setting for cross-genre authorship attribution. The foreground set addresses the difficulty of obtaining cross-genre samples from the same author in naturally occurring Web data. To highlight the cross-genre variation, we present sample texts from a foreground author in two distinct genres in Table 1.

In HIATUS's HRS1 and HRS2 datasets, the queries are always drawn from the foreground collection. For each query, the corresponding needle document(s) are guaranteed to be sampled from genres *different* from from that of the query. This design enforces a strict evaluation of cross-genre performance. The needles in addition to the background documents comprise the candidate pool. Details about the test collection is provided in Appendix B.

3.3 Metrics

We assess system performance with established ranking metrics commonly used in IR. For all ex-

[3] Words are defined as white-space delimited spans.

[4] The upper bound helps eliminate bot-generated content.

[5] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

[6] https://www.iarpa.gov/research-programs/hiatus

[7] https://www.iarpa.gov/

[8] All test data and scripts to create splits used in this work are publicly available through https://www.iarpa.gov/research-programs/hiatus

periments, we report Success@k at $k \in 8, 100$[9], as well as Mean Reciprocal Rank at 20 (MRR@20).

Success@k measures the proportion of queries for which at least one needle, i.e. a candidate document written by the same author as the query, is ranked among the top k results. The mean reciprocal rank at k (MRR@k) computes the average reciprocal rank of the first correct candidate, considering only the top k ranked documents per query.

3.4 Baseline Models

We consider two broad categories of models: IR models and fine-tuned authorship-specific models.

IR Baselines: IR models provide a reference point for quantifying the improvements gained through task-specific adaptation.

- BM25: We use the BM25 relevance score as a measure of authorship similarity. Since BM25 relies solely on lexical matching of content-bearing words, we show that it is ill-suited for cross-genre AA, where lexical overlap, especially of content-bearing words, between documents by the same author may be minimal.

- SBERT: The authorship similarity score is derived from the cosine between the SBERT (Reimers and Gurevych, 2020) embeddings of the query and the candidate documents. Similar to BM25, we show that SBERT underperforms on the cross-genre AA task because the query and relevant candidate documents exhibit substantial topical dissimilarity.

Authorship-specific models: In addition to the IR models, we consider SADIRI (Fincke and Boschee, 2024) and LUAR (Rivera-Soto et al., 2021); models which are explicitly fine-tuned for the AA task.

- LUAR: LUAR (Rivera-Soto et al., 2021) is a RoBERTa-base (Liu et al., 2019) model fine-tuned for AA. Similar to our approach, LUAR also targets ranking-based evaluation and is trained using the contrastive loss objective (Eq. 2). However, LUAR's training curriculum exhibits two key shortcomings: (1) it does not leverage hard negative mining, which is critical for discriminative representation learning, and (2) it is not explicitly optimized for the cross-genre setting.

- SADIRI: SADIRI (Fincke and Boschee, 2024) fine-tunes a RoBERTa-large model for AA. It addresses the two primary limitations of LUAR by incorporating hard negative mining and carefully curating the training data to promote robust cross-genre performance. Among all baselines considered, SADIRI is the strongest, representing the current state-of-the-art in cross-genre AA.

3.5 Implementation Details

Baselines: For the BM25 experiments, we use the standard hyperparameters: $k_1 = 0.25$ and $b = 0.75$. For SBERT, we use the all-mpnet-base-v2[10] model. We use the settings proposed in (Fincke and Boschee, 2024) for SADIRI (Appendix C).

SADIRI-V2 *retriever*: Our proposed retriever improves upon SADIRI by replacing the RoBERTa-large bi-encoder with a decoder-only LLM backbone. We use SADIRI 's default hyperparameters, including hard-negative batching, with the following key efficiency-focused adjustments: (1) Parameter-efficient tuning: Instead of updating all the LLM weights, we only train LoRA parameters (Hu et al., 2021). (2) Reduced batch size: We use smaller batches (e.g., 16 authors per batch for Qwen[11] and Mistral-Nemo-Base-2407[12] vs. 74 in SADIRI). (3) Fewer training epochs: We trained for only a single epoch. (4) Efficient embedding caching: SADIRI 's hard-negative batching necessitates caching the embeddings of the entire training set. However, with LLMs, this naive approach leads to out-of-memory issues which we overcome by introducing a lightweight caching mechanism optimized for memory efficiency. For additional details, please refer to Appendix C

We investigate the effect of model scaling on AA performance using Qwen3 models of varying sizes, with our largest experiment conducted on a 12-billion-parameter Mistral-Nemo-Base-2407 model.

SADIRI-V2 *reranker*: Like our retriever, the SADIRI-V2 reranker also finetunes LLMs by using the cross-genre-targeted training collection described in Section 3.1. For each author a in the training collection, we randomly label one of the authored documents as the query (d_q) and the other

[9] Success@8 is the official evaluation metric for the HIATUS task.

[10] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

[11] https://huggingface.co/collections/Qwen/qwen3-67dd247413f0e2e4f653967f

[12] https://huggingface.co/mistralai/Mistral-Nemo-Base-2407

Author	Type	Genre	Text
A	query	Instructions	`How to Make a Curtain Tie-Back\nIntroduction: Curtain Tie-Backtie-backs are a fantastic ...`
A	needle	News	`Earthquakes in Turkey and Syria\nAn earthquake of the degree of 7.8 recently occurred in ...`
B	distractor 1	Instructions	`How to make a homemade rug\nThe homemade rug is time costly but with determination, you are ...`
C	distractor 2	News	`Media Reportage on Conflict (Russia & Ukraine)\nConflict simply means ...`

Table 1: Example documents (first few sentences) from HIATUS's HRS1 collection illustrating the challenge of cross-genre authorship attribution. The query and needle come from different genres and topics, requiring the system to capture authorial style rather than topical overlap. At the same time, the system must reject distractors that are topically similar to either the query or the needle.

as the positive candidate (d_q^+). We then sample m other documents from the training corpus not authored by a to form the negative set D^-. Thus, in total, each d_q is compared against a total of $(1+m)$ candidate documents. For all our experiments, we fix the number of negatives to $m = 12$. An input instance to the LLM consists of a query and a candidate document concatenated using a rarely occurring character as the delimiter[13]. Both the query and candidate are tokenized independently with a maximum length of 512 tokens, resulting in a combined context length of 1,024 tokens (excluding special tokens). For all reranker experiments, we use Mistral-Nemo-Base-2407 as the backbone LLM and optimize only the LoRA parameters during training (See Appendix C).

As described in Section 2.3, there are three categories from which negative documents can be sampled: $n(d_q)$, $n(d_q^+)$ and $n(r)$. These categories rely on a measure of closeness to the query (d_q) and the positive document (d_q^+). In our experiments, we define closeness using SBERT-based[14] cosine similarity. When negatives are sampled from multiple categories, we sample uniformly across categories, e.g., when sampling from $n(d_q) + n(d_q^+) + n(r)$, we select $m/3$ negatives from each category.

Training a reranker is computationally more expensive than training the first stage retriever. For the retriever, incorporating n authors into a batch requires only $2n$ individual documents as inputs to the transformer. By contrast, for n query authors in a batch, the reranker requires $n(1+m)$ query–candidate pairs, which translates to $(1+m)/2$ additional transformer inputs per author compared to the retriever. Furthermore, it must process both the query and the candidate document within the same sequence, effectively doubling the context length relative to the retriever. Given the significantly higher computational cost, we train the reranker on only 10% of the query-authors used to train the retriever.

Since the reranker has a quadratic run-time complexity of $|\mathcal{Q}| \times |\mathcal{C}|$, to balance effectiveness with efficiency, in all our experiments, at inference time, the reranker is applied only to the top-k ($k = 100$) candidates retrieved by the first-stage retriever.

4 Results

Table 2 presents the performance of retrieval models for cross-genre AA. BM25 and SBERT are incapable of discriminating authorship style. This demonstrates the challenge of AA where models solely relying on content (either lexical or semantic) are unable to effectively perform attribution. LUAR, which was trained specifically for AA, performs better than these baselines, but still lags behind SADIRI and SADIRI-V2 due to architectural and training limitations. In particular, LUAR relies on RoBERTa-base, uses truncated inputs (32 tokens per document and aggregates tokens across 16 documents for each author), and does not optimize for cross-genre or employ hard-negative batching.

SADIRI is the current state-of-the-art model for cross-genre AA. It overcomes LUAR's limitations by targeting cross-genre performance, training efficiently using hard-negative batching and training with maximum context length of 512 tokens using

[13]We use Unicode character U+2980 which never appeared in our training set.

[14]https://huggingface.co/sentence-transformers/all-mpnet-base-v2

Framework	Model / Params	HRS1 / English Long			HRS2 / English Medium		
		Success@8	Success@100	MRR@20	Success@8	Success@100	MRR@20
BM25	—	0.0	0.0	0.0	0.0	5.7	0.0
SBERT	all-mpnet-base-v2 / 0.1B	0.2	3.1	0.3	0.7	3.4	0.2
LUAR	RoBERTa-base / 0.1B	7.2	25.3	3.7	18.2	49.5	12.3
SADIRI	RoBERTa-large / 0.3B	26.0	62.8	14.8	34.0	58.2	18.6
SADIRI-V2	Qwen3 / 0.6B	27.5	67.3	16.8	30.6	64.2	19.4
SADIRI-V2	Qwen3 / 1.7B	32.2	77.5	18.2	37.1	65.7	23.3
SADIRI-V2	Qwen3 / 4B	32.7	78.7	18.7	46.2	74.7	30.6
SADIRI-V2	Qwen3 / 8B	35.7	83.8	21.9	51.7	77.3	33.4
SADIRI-V2	Mistral / 12B	**42.1**	**84.9**	**24.0**	**59.9**	**84.2**	**45.6**

Table 2: SADIRI-V2's cross-genre AA performance of retrieval-based models on the HRS1 and HRS2 datasets. Our results show that scaling model size consistently improves performance, with the 12B Mistral model surpassing previous state-of-the-art, SADIRI, by 16.1 Success@8 points on HRS1 and by 25.9 points on HRS2.

RoBERTa-large. Building on SADIRI, our SADIRI-V2 retrievers incorporate LLM backbones. The Qwen family of models shows a clear trend: larger models yield consistent improvements, with performance rising from 0.6B to 8B parameters. The biggest model we build, Mistral, has 12B parameters, yielding an absolute gain of 16.1 Success@8 points for HRS1 and 25.9 points for HRS2.

A key observation across all models and datasets is the consistently large gap between Success@8 and Success@100. For instance, with the Mistral model, this gap is 42.8 points $(84.9 - 42.1)$ on the HRS1 dataset. In other words, while the retriever often fails to place the correct document within the top-8, it frequently succeeds in ranking it within the top-100. This motivates our use of a reranker on the top-100 candidates, with the goal of narrowing the gap between Success@8 and Success@100.

Table 3 reports SADIRI-V2's reranker performance. For each query document, we reorder the top-100 candidates retrieved by our strongest retriever, SADIRI-V2/Mistral (the retriever performance is shown in the first row of Table 3). All rerankers are built on top of a pretrained Mistral model[15]. As discussed in Section 3.5, when training the reranker, in addition to the single positive document, we compare each query document against $m = 12$ negative documents. The distribution of these m negative documents ranges from $n(r)$, where all negatives are randomly sampled, to $n(r) + n(d_q) + n(d_q^+)$, where equal number (4 when $m = 12$) of negatives is drawn from each of $n(r)$, $n(d_q)$, and $n(d_q^+)$.

We observe that the strategy used to sample negative training documents has a significant impact on the reranker performance. Specifically, we see that the performance of the reranker is best when negative documents are drawn from the distribution $n(r) + n(d_q) + n(d_q^+)$, providing a significant improvement of Success@8 compared to the retriever alone by 6.2 points for HRS1 and 8.5 for HRS2.

Sampling only from $n(d_q)$ results in the worst performance: From Table 3, it is clear that selecting negatives only around the query, i.e. selecting from $n(d_q)$, yields the weakest performance across datasets and metrics, even worse than the retriever itself. The reason is rooted in how training data are constructed: d_q and d_q^+ are deliberately chosen to be semantically dissimilar to enforce cross-genre robustness (see Section 3.1). When the reranker is trained to score d_q^+ above negatives sampled only from $n(d_q)$, the optimization reduces to learning an anti-topic heuristic: documents semantically close to d_q are always scored low, while those semantically farther away are scored high. Thus, this failure mode is an inherent consequence of adapting the reranker to the cross-genre AA setting.

5 Related Work

LLMs for authorship analysis: (Hung et al., 2023; Huang et al., 2024) reported that LLMs are capable of determining whether or not two documents are written by the same author (the task of authorship verification). However, their methodology has three key limitations: 1) they did not test on cross-genre scenario and so their conclusion is likely to be confounded by LLM's ability to pick that same authors tend to write about similar topics, 2) they tested on datasets such as Enron emails (Klimt and Yang, 2004) which are very likely to be already present in most LLM's pretraining dataset, and 3) they solely explored LLM's zero-shot ability via

[15] https://huggingface.co/mistralai/Mistral-Nemo-Base-2407

Framework	Negative sampling strategy	HRS1 / English Long			HRS2 / English Medium		
		Success@8	Success@100	MRR@20	Success@8	Success@100	MRR@20
Retriever	-	42.1	84.9	24.0	59.9	84.2	45.6
Reranker	$n(r)$	40.2	-	26.9	66.4	-	49.1
	$n(d_q)$	<u>34.8</u>	-	<u>17.7</u>	<u>58.8</u>	-	<u>30.6</u>
	$n(d_q^+)$	44.1	-	27.6	66.7	-	**54.2**
	$n(r) + n(d_q)$	47.0	-	30.8	66.7	-	51.6
	$n(r) + n(d_q^+)$	44.0	-	27.7	67.4	-	50.9
	$n(d_q) + n(d_q^+)$	46.6	-	30.6	68.0	-	53.5
	$n(r) + n(d_q) + n(d_q^+)$	**48.3**	-	**31.1**	**68.4**	-	53.1

Table 3: SADIRI-V2's cross-genre AA performance of reranker-based models on the HRS1 and HRS2 datasets. We take the output of our strongest reteiver (Mistral / 12B) and rerank its top-100 retrieved candidate documents. Highest and lowest values under each column are in bold and underlined respectively. The table shows the variation in reranker performance across different negative document sampling strategies. Sampling negatives around the query ($n(d_q)$) results in the worst performance, scoring even below the retriever. $n(r) + n(d_q) + n(d_q^+)$ performs the best and improves over the retriever by 6.2 Success@8 points on HRS1 and 8.5 points on HRS2.

prompting. We address all three limitations; we target cross-genre AA, test on the HIATUS dataset in which the foreground documents are solely written for the purpose of IARPA's HIATUS program and are less likely[16] to have already been included in LLM's pretraining collection, and finally, we finetune LLMs to adapt it for the AA task.

Retrieve-and-rerank for AA: Previous work has framed authorship attribution as a ranking task (Rivera-Soto et al., 2021; Fincke and Boschee, 2024), where the candidate pool may include tens of thousands of documents, similar to large-scale settings in information retrieval (IR). Prior ranking-based approaches to authorship analysis, however, have relied solely on the retrieval component (Rivera-Soto et al., 2021; Fincke and Boschee, 2024). We extend this line of work by developing an effective reranker to significantly improve the system's end-to-end performance.

Cross-genre authorship attribution: Previous works have explored methods to ensure authorship verification and analysis models encode linguistic style alone and not content/topic/symmantics (Halvani et al., 2019; Wegmann et al., 2022; Fincke and Boschee, 2024). We follow SADIRI's approach by training on pairs of an author's most semantically dissimilar documents to reduce topic bias. Additionally, to the best of our knowledge, we are the first to address the challenge of building an effective cross-encoder/reranker for cross-genre AA[17].

Negative document sampling in IR: Training IR models benefits substantially from the inclusion of hard negatives (Xiong et al., 2020; Gao et al., 2021a). Typically, these negatives are selected to be close to the query, without explicitly considering their distance from the positive document (Xiong et al., 2020). Some studies (Yang et al., 2024; Meghwani et al., 2025), however, incorporate the positive document into the sampling process to reduce false negatives; documents labeled as irrelevant to the query but that are in fact relevant which is a common issue in IR training data.

6 Conclusion

This work extends the retrieve-and-rerank framework, a pipeline that is canonically used in IR, to the field of cross-genre AA. Leveraging LLMs for retievers enables us to surpass previous state-of-the-art, SADIRI, by a significant margin (16.1 Success@8 points on HRS1 and 25.9 points on HRS2). We find that building an effective reranker for cross-genre AA is far from straightforward: when hard negatives are sampled using the IR-style strategy of placing them near the query, performance not only degrades, but the reranker scores below the retriever itself. Our cross-genre targetted reranker delivers substantial gains (6.2 Success@8 points on HRS1 and 8.5 points on HRS2), further boosting performance beyond our already strong retriever. Taken together, our two-stage retrieve-and-rerank pipeline improves cross-genre HRS1 and HRS2 scores over the previous state-of-the-art, SADIRI, by 22.3 and 34.4 Success@8 points, respectively.

[16]HIATUS prohibits the use of its data in closed-source models such as GPT.

[17]Previous works (Wegmann et al., 2022; Fincke and Boschee, 2024) developed content-controlled authorship analysis models, but they focused on bi-encoders/retrievers rather than cross-encoders/rerankers.

Limitations

Improvements made in this work are based on using larger models than previously used for AA. Training and inference with these LLMs is computationally expensive. For instance, the Mistral-12B retriever is approximately 20× slower than RoBERTa-large during training and about 6× slower at inference. Because the reranker jointly encodes each query-candidate pair, its computational expense is higher than the retrievers. To compensate for this, we train the reranker on only 10% of our training collection and rerank only the top-100 retrieved documents.

Due to hardware constraints, Mistral-Nemo-Base-2407 is the largest model (in terms of number of parameters) that we trained. Whether the performance of the attribution system further improve upon using even larger models remains an open question.

Selecting negative documents for the reranker requires one to define a measure of closeness. In this work, we define closeness based on document embeddings derived from an SBERT model. In IR, alternative measures of closeness (such as BM25 scores or retriever-derived embeddings (Gao et al., 2021b)) are also widely used, and exploring these remains an interesting direction for future work.

Acknowledgements

References

Sarkhan Badirli, Mary Borgo Ton, Abdulmecit Gungor, and Murat Dundar. 2021. Open set authorship attribution toward demystifying victorian periodicals. In *Document Analysis and Recognition – ICDAR 2021*, pages 221–235, Cham. Springer International Publishing.

Georgios Barlas and Efstathios Stamatatos. 2020. Cross-domain authorship attribution using pre-trained language models. In *Artificial Intelligence Applications and Innovations*, pages 255–266, Cham. Springer International Publishing.

Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. 2016. Training deep nets with sublinear memory cost.

Steven Fincke and Elizabeth Boschee. 2024. Separating style from substance: Enhancing cross-genre authorship attribution through data selection and presentation.

Luyu Gao, Zhuyun Dai, and Jamie Callan. 2021a. Rethink training of bert rerankers in multi-stage retrieval pipeline. In *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28 – April 1, 2021, Proceedings, Part II*, page 280–286, Berlin, Heidelberg. Springer-Verlag.

Luyu Gao, Zhuyun Dai, and Jamie Callan. 2021b. Rethink training of bert rerankers in multi-stage retrieval pipeline.

Oren Halvani, Christian Winter, and Lukas Graner. 2019. Assessing the applicability of authorship verification methods. In *Proceedings of the 14th International Conference on Availability, Reliability and Security*, ARES '19, New York, NY, USA. Association for Computing Machinery.

J. Edward Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *ArXiv*, abs/2106.09685.

Baixiang Huang, Canyu Chen, and Kai Shu. 2024. Can large language models identify authorship? *ArXiv*, abs/2403.08213.

Chia-Yu Hung, Zhiqiang Hu, Yujia Hu, and Roy Ka-Wei Lee. 2023. Who wrote it and why? prompting large-language models for authorship verification.

Abraham Israeli, Shuai Liu, Jonathan May, and David Jurgens. 2025. The million authors corpus: A cross-lingual and cross-domain Wikipedia dataset for authorship verification. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 25997–26017, Vienna, Austria. Association for Computational Linguistics.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

William B. Johnson and Joram Lindenstrauss. 1984. Extensions of lipschitz mappings into hilbert space. *Contemporary mathematics*, 26:189–206.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

Bryan Klimt and Yiming Yang. 2004. The enron corpus: a new dataset for email classification research. In *Proceedings of the 15th European Conference on*

Machine Learning, ECML'04, page 217–226, Berlin, Heidelberg. Springer-Verlag.

Robert Layton, Paul Watters, and Richard Dazeley. 2010. Authorship attribution for twitter in 140 characters or less. In *2010 Second Cybercrime and Trustworthy Computing Workshop*, pages 1–8.

Robert Leonard, Juliane Ford, and Tanya Christensen. 2017. Forensic linguistics: Applying the science of linguistics to issues of the law. *Hofstra law review*, 45.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2024. Fine-tuning llama for multi-stage text retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 2421–2425, New York, NY, USA. Association for Computing Machinery.

Hansa Meghwani, Amit Agarwal, Priyaranjan Pattnayak, Hitesh Laxmichand Patel, and Srikant Panda. 2025. Hard negative mining for domain-specific retrieval in enterprise systems.

Frederick Mosteller and David L. Wallace. 1984. Applied bayesian and classical inference : the case of the federalist papers.

Trang Nguyen, Charlie Dagli, Kenneth Alperin, Courtland Vandam, and Elliot Singer. 2023. Improving long-text authorship verification via model selection and data tuning. In *Proceedings of the 7th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 28–37, Dubrovnik, Croatia. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Rafael A. Rivera-Soto, Olivia Elizabeth Miano, Juanita Ordonez, Barry Y. Chen, Aleem Khan, Marcus Bishop, and Nicholas Andrews. 2021. Learning universal authorship representations. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 913–919, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jacob Tyo, Bhuwan Dhingra, and Zachary Lipton. 2022. On the state of the art in authorship attribution and authorship verification. *ArXiv*, abs/2209.06869.

Lorenzo Valla. 1922. *Discourse on the Forgery of the Alleged Donation of Constantine*. Yale University Press, New Haven.

Andrew Wang, Cristina Aggazzotti, Rebecca Kotula, Rafael Rivera Soto, Marcus Bishop, and Nicholas Andrews. 2023. Can authorship representation learning capture stylistic features? *Transactions of the Association for Computational Linguistics*, 11:1416–1431.

Anna Wegmann, Marijn Schraagen, and Dong Nguyen. 2022. Same author or just same topic? towards content-independent style representations. In *Proceedings of the 7th Workshop on Representation Learning for NLP*, pages 249–268, Dublin, Ireland. Association for Computational Linguistics.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *ArXiv*, abs/2007.00808.

Zhen Yang, Zhou Shao, Yuxiao Dong, and Jie Tang. 2024. Trisampler: a better negative sampling principle for dense retrieval. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'24/IAAI'24/EAAI'24. AAAI Press.

A Training data

We train all our models on data scraped from 8 web sources. To encourage our AA models to rely on authorial style and not on topic, we include the two most topically dissimilar documents by each training author and include the document-pair only if their SBERT similarity is less than a threshold. Across all experiments, similarity was computed as the dot product between document embeddings generated by the all-mpnet-base-v2[18] SBERT model and threshold of 0.2 was used to be consistent with SADIRI. The training data is processed by removing personally identifiable information (PII). Specifically, identifiers pertaining to person names, telephone numbers, email addresses and ip-addresses are replaced by to PERSON, PHONE_NUMBER, EMAIL_ADDRESS and IP_ADDRESS respectively. Note that removing PII is not only crucial for preserving privacy but also essential for fair AA evaluation, since such identifiers can provide trivial author-specific cues, allowing models to bypass learning stylistic features. In Table 4, we list all the data sources along

[18] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

with the number of documents and authors which we used in training of all the SADIRI and SADIRI-V2 models.

B Test data

We benchmark our retrieve-and-rerank AA system on the HRS1 and HRS2 English HIATUS datasets (HRS stand for **H**IATUS **R**esearch **S**et). These datasets are publicly available through the HIATUS website: https://www.iarpa.gov/research-programs/hiatus. Documents in HRS1 and HRS2 are drawn from five different genres. The number of documents in each genre for both HRS1 and HRS2 are shown in Tables 5 and 6 respectively. For each genre, we provide the number of documents in the foreground and the background collection. Also, the websites from which the background documents are scraped are also shown. Distribution of the number of documents in the foreground and background collections, along with unique number of authors is shown in Table 7.

The query documents in HRS1 and HRS2 are always drawn from the foreground collection. To construct the query pool, we first sample a fraction of foreground authors. For each selected author, one of the five genres is randomly chosen, and all of that author's documents in the chosen genre are included as queries. The same author's documents in the remaining genres are placed in the candidate pool, ensuring evaluation under a cross-genre setting. In addition, all background documents are added to the candidate collection.

Because the number of foreground authors and documents is limited (see Table 7), the sampling process typically yields fewer than 100 query documents. To reduce variability, we construct $s = 4$ distinct query–candidate splits, each generated with a different random seed (seed 0, 1001, 2001, 3001) when sampling query authors and genres. System performance is then averaged across these splits, and the results are reported in Tables 2 and 3. The exact number of queries and candidate documents in each split is provided in Table 8.

C Training configuration

Retriever hyperparameters: All SADIRI and SADIRI-V2 models are optimized using the Adam optimizer (Kingma and Ba, 2017) with a learning rate of $1e-5$ and a gradient accumulation step of 1. We set $\tau = 0.01$. To be consistent with (Fincke and Boschee, 2024), we train the SADIRI models for 4 epochs with per-gpu batch-size of 74 authors. To compensate for a larger parameter footprint when training LLMs, the SADIRI-V2 models are only trained for a single epoch with per-gpu batch-size of 16 authors. All retrievers are trained on a single node with 4 RTX-A6000 gpus. All parameters are updated in bfloat16 and we use gradient checkpointing (Chen et al., 2016) to save memory while training. It takes roughly 15 hours to train our largest model (Mistral-12B).

Reranker hyperparameters: All rerankers are finetuned versions of Mistral-Nemo-Base-2407[19] and are trained using the Adam optimizer (Kingma and Ba, 2017) with a learning rate of $1e-4$. We set $\tau = 1.0$. Each per-gpu training batch contains 1 query document. For each query document, in addition to the single positive document, we select a total of $m = 12$ negative documents. We use gradient accumulation of 10 steps. All rerankers are trained on a single node with 4 RTX-A6000 gpus. All parameters are updated in bfloat16 and we use gradient checkpointing (Chen et al., 2016) to save memory while training. It takes roughly 60 hours to train each of the rerankers.

All retriever and reranker experiments involving Qwen3 and Mistral update only the LoRA parameters during training. We leverage PEFT's[20] implementation of LoRA. Table 9 shows the LoRA configuration that we used for all our experiments.

In addition to the LoRA parameters, we also train the projection matrix, W, in Eq. 1 and 3.

SADIRI's hard negative batching approach: SADIRI proposed to train the retriever by constructing training batches of similar authors. This ensures that the in-batch negatives used in the contrastive loss are hard, providing better statistical pressure for the retriever to weed out the negatives in search of the positive document. The exact algorithm for this approach is shown in Algorithm 1. Consistent with SADIRI, we set the scaling factor $s = 3.5$ across all our experiments. At the start of each epoch, the document embeddings from the current state of the retriever are used to cluster the entire training collection. Subsequently, the embeddings are clustered using K-means (via Faiss (Johnson et al., 2019)). Thus, this approach is memory intensive, requiring the embeddings of the entire training data to be loaded into GPU memory. Since

[19] https://huggingface.co/mistralai/Mistral-Nemo-Base-2407

[20] Version 0.17.1 of `https://github.com/huggingface/peft` installed via pip.

source	description	# docs / # authors
RealNews	news stories	191,020 / 95,510
BookCorpus	full-length books	66,786 / 33,393
Reddit	reddit.com entries	55,014 / 27,507
GoodReads	book reviews	34,010 / 17,005
Amazon	reviews on amazon.com	17,476 / 8,738
Gmane	newsgroups	16,396 / 8,198
WikiDisc	Wikipedia editorial discussions	13,724 / 6,862
PubMed	medical journal articles	692 / 346

Table 4: Training data sources along with the number of document and authors selected from each source.

Genre description	Sources	# Foreground docs	# Background docs
Board game reviews	boardmangeek.com	212	2,432
Citizens journalism	globalvoices.org	140	5,453
Instructions	instructables.com	133	8,534
Literature forums	stackexchange.com	199	9,641
STEM forums	stackexchange.com	171	11,024

Table 5: Distribution of documents across genres in HIATUS's HRS1 test set.

Genre description	Sources	# Foreground docs	# Background docs
Posts on pets	reddit.com	112	1,298
Personal tales and experiences	reddit.com	138	1,542
Product reviews	-	121	0
Obituary	multiple†	132	3,625
Movie reviews	letterboxd.com	128	4,377

Table 6: Distribution of documents across genres in HIATUS's HRS2 test set. †The obituary posts are scraped from ballardsunderfuneral.com, dogwoodcremationcare.com, englandfuneralhome.com, pierfuneralhome.com, and snyderfuneralhome.com.

HRS	# Foreground docs/authors	# Background docs/authors
HRS1	855/114	37084/16313
HRS2	631/37	10842/2581

Table 7: Distribution of documents and unique authors in forground and background collection of HRS1 and HRS2.

Sampling seed	HRS1		HRS2	
	Query docs	Candidate docs	Query docs	Candidate docs
0	119	34370	72	7357
1001	112	34157	74	7393
2001	104	33150	76	7464
3001	129	31139	75	7365

Table 8: Number of query and candidate documents for HRS1 and HRS2 across different sampling seeds used to construct the query/candidate splits.

Property	Value
r	16
lora_alpha	$r \times 2$
target_modules	{q,v,k,o,gate,up,down}_proj
lora_dropout	0.05
bias	none
task_type	FEATURE_EXTRACTION

Table 9: PEFT's LoRA configuration which was used for all retriever and reranker experiments involving Qwen3 and Mistral.

LLMs result in much larger document embeddings than RoBERTa-large, it becomes necessary to handle this memory requirement efficiently.

Efficient implementation using random projection: The clustering depends on cosine similarity of document embeddings, $\cos(v(d_1), v(d_2)) = v(d_1)^T v(d_2)/\|v(d_1)\|\|v(d_2)\|$. We use this fact to reduce memory footprint by applying random matrix projection to each document embedding. This is motivated by the fact that (Johnson and Lindenstrauss, 1984) asserts that random projections preserve inner products. Intuitively, let $r(d) = Rv(d_1)$ where $r \in R^K$ is the reduced dimensional vector that we get by projecting $v \in R^D$ using the random projection matrix $R \in R^{K \times D}$, $K < D$. Then

$$\begin{aligned} r(d_1)^T r(d_2) &= (Rv(d_1))^T (Rv(d_1)) \\ &= v(d_1)^T R^T R v(d_1). \end{aligned}$$

If each element in R is drawn in an i.i.d from a univariate zero-mean distribution, then $R^T R \approx K\mathbb{1}$. With this assumption, we get $r(d_1)^T r(d_2) \approx K v(d_1)^T v(d_1)$, that is, the dot product of the projected vectors is approximately the same as the dot product of the original vectors (up to a scaling constant, K). Since the scaling factor gets absorbed in normalization when calculating the cosine distance, we get $\cos(r(d_1), r(d_2)) \approx \cos(v(d_1), v(d_2))$. For our experiments, we set $K = D//3$.

Algorithm 1 SADIRI's batching approach at the start of each epoch

Require: Training set of N authors, each with 2 documents: $\mathcal{D} = \{d_{i1}, d_{i2}\}_{i=1}^{N}$
Require: Embedding of all $2N$ documents ▷ Model output from previous epoch (For epoch 0, take output from the untrained model.)
Require: Batch size b authors, clustering factor s
Ensure: Similar author in the same batch

```
Cluster all 2N documents into k = b·s clusters
for each author a_i do ▷ Assign each author to a unique cluster
    if documents d_i1, d_i2 are in different clusters then
        Keep the document in the larger cluster
    end if
end for
for clusters C_j with > b authors do ▷ Split/merge to respect batch size
    Redistribute authors to smaller clusters
end for
Randomly sample s clusters to form each batch
```