

Loci Similes: A Benchmark for Extracting Intertextualities in Latin Literature

Julian Schelb[†], Michael Wittweiler[◊], Marie Revellio^{‡*},
Barbara Feichtinger[‡], and Andreas Spitz^{†*}

[†]Department of Computer and Information Science, University of Konstanz

[‡]Department of Latin Philology, University of Konstanz

[◊]Institute of Archaeology, Classical Philology and Ancient Studies, University of Zurich

{firstname.lastname}@uni-konstanz.de

{firstname.lastname}@uzh.ch

Abstract

Tracing connections between historical texts is an important part of intertextual research, enabling scholars to reconstruct the *virtual library* of a writer and identify the sources influencing their creative process. These intertextual references manifest in diverse forms, ranging from direct verbatim quotations to subtle allusions and paraphrases disguised by morphological variation. Language models offer a promising path forward due to their ability to capture semantic similarity beyond lexical overlap. However, the development of new methods for this task is held back by the scarcity of standardized benchmarks and easy-to-use datasets. We address this gap by introducing *Loci Similes*, a benchmark for Latin intertextuality detection comprising a curated dataset of ~176k text segments and 1,490 expert-verified intertextual references, including 945 from an existing dataset. Using this data, we establish baselines for retrieval and classification of intertextualities with both lexical methods and pretrained encoder language models.

1 Introduction

Identifying intertextual references between documents is an important task in classical philology, as it reveals how later works engage with earlier texts and traditions. For centuries, scholars detected intertextual references by relying on memory and the manual collation of *Loci Similes*, i.e., parallel passages that exhibit lexical, semantic, or thematic resemblance. Although digitization has augmented this process through lexical search tools, most approaches still depend on exact n-gram matching or heuristic filtering (Schropp et al., 2024a). This limits discovery rates in ancient texts, where intertextuality not only manifests itself as verbal quotation, but also as subtle allusion, paraphrase, or thematic variation (Manjavacas et al., 2019; Gong

Example of Historical Text Reuse

Source: Virgil, *Aeneid* 2.774

Context: Aeneas is terrified when the ghost of his wife Creusa appears to him during the burning of Troy.

“... obstipui, steteruntque comae et uox faucibus haesit.”

(... I was stupefied, my hair stood on end, and my voice stuck in my throat.)

Reuse: Jerome, *Epistula* 130.5.5

Context: A family’s shocked reaction to Demetrias’s vow of Christian virginity.

“Haesit uox faucibus et inter ruborem atque pallorem metumque ac laetitiam cogitationes uariae mutabantur.”

(The voice stuck in their throat, and between blushing and pallor, fear and joy manifold thoughts kept shifting.)

Figure 1: **Example of intertextual reference.** Reuse of a classic Virgilian phrase for speechlessness by Jerome. While retaining the semantic core, the author alters the word order to adapt the expression to a different context.

et al., 2025), often complicated by orthographic volatility (Miller et al., 2025).

Recovering such text reuse is not merely a matter of identifying sources. It facilitates research on broader cultural-historical phenomena (Tangherlini and Chen, 2024). It supports work on reception and cultural hybridization in Late Antiquity, where pagan texts persist as the rhetorical substrate of elite writing while being recontextualized within emerging Christian discourse. Classical forms remain recognizable even as their functions shift toward Christian meaning-making, visible in computational stylometry (Gorman and Gorman, 2016; Di Bernardo et al., 2021) and semantic motifs. Citation patterns suggest how Christian authors do not abandon classical texts and their cultural contexts, but reuse their language, redirecting its meanings within Christian frameworks.

*These authors contributed equally as project lead

A case in point is the Church Father Jerome. Recent digital-hermeneutic studies have begun to map his “micro-quotations” (Schropp et al., 2024a), yet the semantic breadth of his reuse remains a challenge. When Jerome alludes to the Augustan poet Virgil, he often retains the semantic core of a hexameter verse while altering its word order or syntax to suit his Christian prose context. In Jerome’s writings, the general tension between his pagan *paideia* and Christian discourse emerges with particular clarity in the details of how he quotes classical pagan sources and adapts them for his own texts (see Figure 1). Since canonical authors such as Virgil were deeply embedded in the educational curriculum, quoting them often served as a shorthand for shared cultural memory. Late Antique Christian writers like Jerome inherit this repertoire, but their reuse frequently reframes pagan language within Christian contexts, making citation patterns a measurable trace of shifting cultural authority.

Beyond extraction, systematically mapping these references allows scholars to reconstruct the “virtual library” available to an author and to gain insight into which sources most strongly shaped their writing. Analyzing how these references cluster, from explicit citation to subtle echo, further helps refine theoretical definitions of intertextuality and probe the rhetorical motivations behind text reuse. Although some resources exist (Burns et al., 2021; Toyin et al., 2026), language-model-based detection of intertextual references remains constrained by a lack of standardized benchmarks and datasets.

In this paper, we take a step towards addressing this gap by introducing *Loci Similes*, a benchmark designed to enable researchers to systematically compare and evaluate computational approaches for intertextuality detection.¹ We contribute:

- **Curated benchmark dataset** of ~176k Latin text segments, partitioned into query and source corpora, accompanied by a ground-truth dataset of 1,490 expert-verified intertextual references.
- **Evaluation framework** that aligns more closely with the practical constraints of philological workflows.
- **Baseline results** for retrieval, classification, and a retrieve-and-rerank pipeline, serving as a foundation for future comparisons.

¹Dataset, fine-tuned models, and code are available at <https://github.com/julianschelb/locisimiles>.

2 Related Work

2.1 Intertextuality Detection in Latin

Work on Latin intertextuality ranges from surface-level matching to increasingly semantic representations. Revellio (2022) developed a rule-based approach operating directly on surface strings to uncover Virgilian references in Jerome’s letters, and Schropp et al. (2024a) refined this with n-gram matching pipelines and cascaded filters to detect very short references. Such methods capture verbatim and near-verbatim reuse well, but miss semantically related passages with little shared vocabulary.

Word-level embeddings instead capture semantic similarity beyond lexical overlap. Burns et al. (2021) used static Word2Vec models trained on lemmatized text to rank intertextual phrases, evaluated against 945 parallels from Valerius Flaccus’ *Argonautica* (Dexter et al., 2024). Manjavacas et al. (2019) approached allusion detection in Latin sermons as information retrieval using fastText embeddings, and Manjavacas et al. (2020) extended this work on the *Patrologia Latina*, modeling lexical and thematic similarity as separate axes.

Gong et al. (2025) replaced static vectors with contextual LatinBERT (Bamman and Burns, 2020) embeddings, retrieving a target word’s nearest neighbors by cosine similarity to identify allusions in Lucan’s *Pharsalia*. Riemenschneider and Frank (2023b) applied sentence-transformer models (Reimers and Gurevych, 2019) to ancient languages, introducing SPhilBERTa, a trilingual model built on PhilBERTa (Riemenschneider and Frank, 2023a) and fine-tuned on parallel sentences in Ancient Greek, Latin, and English. More recently, Toyin et al. (2026) introduced Gretino, a benchmark for semantic retrieval in Latin, Ancient Greek, and cross-lingual settings. Unlike these resources, *Loci Similes* labels positive references by reference type, allowing verbatim references and allusions to be evaluated separately.

2.2 Text Reuse in Other Languages

Beyond Latin, Kuznetsov et al. (2022) model intertextual relations as a graph, Lau and McManus (2024) use large language models to extract asymmetric intertextuality, and MacLaughlin et al. (2021) show that contextual models outperform lexical baselines. Rosson et al. (2023) built a tool for exploring reuse in Early Modern British texts, whose pipeline (Mahadevan et al., 2025) indexes billions of reuse pairs.

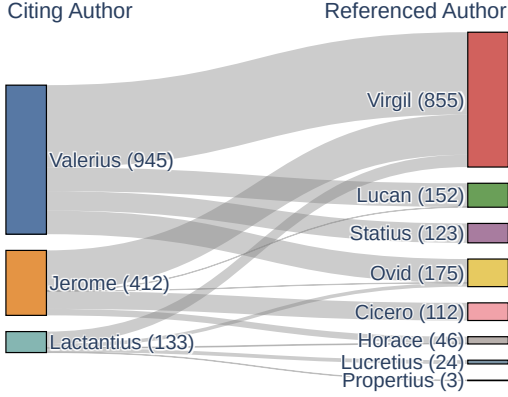


Figure 2: **Distribution of confirmed intertextual references.** Manually verified intertextual references in the annotated dataset by citing author (Jerome, Lactantius, and Valerius Flaccus) and source author (including Virgil, Cicero, and others).

Specialized approaches address low-resource languages. Miller et al. (2025) enhanced reuse detection in Hebrew and Aramaic manuscripts using fastText embeddings, and Gorman and Gorman (2016) modeled the authorial style in Ancient Greek through unsupervised clustering. Sharjeel et al. (2023) addressed cross-lingual reuse in English–Urdu using machine translation.

2.3 Further Related Tasks

Intertextuality detection is related to quote detection (Janicki et al., 2023) and paraphrase identification (Vrbanec and Mestrovic, 2023). However, quote detection targets speech explicitly attributed to a speaker (Petersen-Frey and Biemann, 2024; Zhang and Liu, 2022), and paraphrase detection focuses on semantic equivalence (Wahle et al., 2022). Our task differs in requiring the capture of unmarked allusions and recontextualized echoes at the segment level across unrelated documents.

3 Dataset Construction

While traditional scholarship has documented numerous Latin intertextual references, computational research remains constrained by the lack of standardized benchmarks. Aggregating these references is non-trivial, as the data is dispersed across commentaries, *indices locorum* (“back-of-the-book indexes”), and philological case studies. To address this gap, we curated a corpus of $\sim 176k$ text segments from multiple Latin authors (see Table 1) and a merged ground-truth dataset of 1,490 confirmed intertextual references (see Figure 2).

Author	Segments	Avg. Tokens	Min	Max	Std. Dev.
Query Corpus					
Jerome	74,672	31.13	1	617	22.32
Lactantius	8,444	28.53	1	369	19.48
Valerius	2,138	29.60	1	145	18.10
Total / Avg.	85,254	30.83	1	617	21.97
Source Corpus					
Cicero	54,331	28.97	1	1,377	25.40
Ovid	14,096	26.51	1	432	19.53
Virgil	4,861	29.65	2	353	19.55
Martial	4,114	26.13	2	274	21.07
Lucan	2,952	31.26	2	245	21.83
Statius	2,527	44.08	2	412	40.22
Horace	2,353	33.60	2	195	24.83
Propertius	1,889	22.18	2	267	17.56
Lucretius	1,826	40.95	3	256	27.40
Catullus	809	27.43	2	149	22.08
Tibullus	788	26.12	2	251	20.70
Total / Avg.	90,546	29.17	1	1,377	24.61

Table 1: **Corpus statistics by author.** Breakdown of the dataset into *Query* (citing authors) and *Source* (referenced authors) corpora, showing the number of segments and token statistics for each author.

3.1 Latin Corpus Curation

We compose the corpus of a separate *Query Corpus* and a *Source Corpus*. The former comprises works by the Late Antique authors Jerome and Lactantius, together with Book 1 of Valerius Flaccus’ *Argonautica* added through the subset from Burns et al. (2021), totaling $\sim 85k$ text segments. The latter, which serves as the retrieval target, consists of $\sim 91k$ segments drawn from eleven canonical classical Latin authors, in chronological order: Cicero, Lucretius, Catullus, Virgil, Horace, Tibullus, Propertius, Ovid, Lucan, Martial, and Statius. Text segments correspond to sentences delimited by hard punctuation (periods, exclamation marks, and question marks). A detailed breakdown of segment counts and token statistics for each author is presented in Table 1. The texts were aggregated from three primary digital repositories: *Corpus Corporum*, the *Tesserae Project*, and the *OpenGreekandLatin Project*. For a complete listing of the specific works, critical editions used, and their respective provenance, see Appendix A. Since we exclusively rely on publicly available texts to compile the corpus, we release the corpus files alongside the ground-truth dataset.

3.2 Ground Truth Construction

Our ground-truth dataset combines 545 references curated for *Loci Similes* with 945 labeled references from Valerius Flaccus’ *Argonautica* (Burns et al., 2021), 1,490 in total.

First, we sourced 270 references to Virgil and Cicero in Jerome’s *Epistulae* from the dataset² established by Schropp et al. (2024b), adapting these entries to sentence-level granularity and excluding references disputed by our expert annotators. We added 275 further references, identified with the rule-based *n*-gram matching approach of Schropp et al. (2024a).³ Their pipeline scans for shared, non-contiguous tokens within a fixed window, then refines the candidate pairs through a cascade of filters that remove stopwords, enforce part-of-speech constraints, and exclude generic collocations by embedding similarity. Domain experts then manually evaluated the candidates, retaining only those confirmed as genuine intertextual references.

Second, we incorporated the dataset of Burns et al. (2021), comprising 945 known references between Book 1 of the *Argonautica* and four major Latin epics: Virgil’s *Aeneid*, Ovid’s *Metamorphoses*, Lucan’s *Pharsalia*, and Statius’ *Thebaid*. These references derive from traditional scholarship, namely the commentaries of Spaltenstein (2002), Kleywegt (2005), and Zissos (2008). We aligned their granularity and format with our existing dataset.

Item-level scholarly support extends to nearly the entire benchmark: the 1,215 references adopted from the two existing datasets derive from published commentaries, and 274 of the 275 references added through the *n*-gram pipeline are likewise attested in published editions or commentaries, so that 1,489 of the 1,490 positive references are documented in prior scholarship. The remaining reference is, to our knowledge, a new discovery and will be described in a separate philological publication. For each reference, we record the publication from which it was adopted, so its scholarly support can be verified directly.

3.3 Annotation Process

Constructing the ground truth involved two decisions: establishing whether a candidate pair constitutes an intertextual reference and, if so, assigning its reference type.

Establishing a reference. The pipeline’s filters removed many false positives, but numerous candidates still involved common collocations (e.g., *puncto temporis* “in a moment”) and required manual exclusion. Four experts in Latin literature (two

²<https://doi.org/10.11588/data/FVCULR>

³https://github.com/MWittweiler/citation_analysis

Taxonomy of Intertextual References

1. Verbatim Reference (*cit.*) – Literal Reuse

Near-verbatim reuse sharing at least two congruent lemmas in a coherent context.

- *Marked:* Explicitly attributed to a specific author or source (e.g., *ut ait Cicero...* → “as Cicero says”).
- *Unmarked:* Silent incorporation of exact text strings from other sources.
- *Paraphrased:* Morphological, word-order, or lemma-equivalent variation (*arma uirumque* → *armis uirisque*; *cognoscat* → *cognoscere*).

2. Allusion (*cf.*) – Semantic Similarity

Thematic or stylistic echo without close lexical overlap; synonyms, paraphrase, or shared imagery instead of shared lemmas.

- *Single reference:* Shared distinct vocabulary or imagery recontextualized in a new setting (*infans* → *puer*; *paruulus* → *paruus*).
- *Systemic:* Broad stylistic, rhythmic, or thematic imitation on multiple occasions in the document.

Figure 3: **Annotation scheme.** Following Schropp et al. (2024b), we adopt a two-way reference-type scheme based on *Verbatim References* (*cit.*) and *Allusions* (*cf.*).

pre-PhD, two post-PhD) therefore assessed each of the 275 *n*-gram-derived candidates against the relevant scholarship, confirming references by consensus rather than by independent parallel annotation. Their assessment was guided by three criteria:

1. **Use of Uncommon Vocabulary:** If the lexical overlap consists of rare or marked words, the likelihood increases that these elements were deliberately borrowed.
2. **Attested Frequency:** The *Library of Latin Texts* corpus database⁴ served to determine the frequency of the overlapping expression in Latin literature. Expressions appearing exclusively in the candidate source and target passages were treated as strong indicators of a unique intertextual relationship.
3. **Conduit Function:** Most importantly, if the cited expression contributes semantic, rhetorical, or cultural information from the source text that cannot be derived from the target passage in isolation, thereby implicitly enriching the interpretation, it is viewed as intertextual.

Assigning a reference type. The reference type is assigned with the operational criterion of Schropp et al. (2024b). An established reference is labeled a *verbatim reference* if it contains at least

⁴<http://clt.brepolis.net/llta/Search>

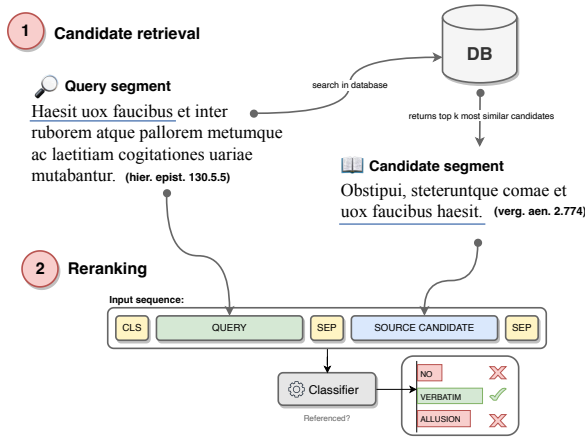


Figure 4: **Two-stage experimental setup.** *Stage 1:* The input text segment acts as a query to retrieve potential candidates from the database. *Stage 2:* To verify the reference, the query and source candidate are concatenated into a single input sequence and passed to a three-class cross-encoder (*no match*, *cit.*, *cf.*).

two matching lemmas in close proximity within text segments delimited by hard punctuation. Close proximity permits at most two intervening words after excluding stopwords such as *et*, *sic*, and *non*. Established references that do not meet these criteria are labeled *allusions*. Given the lemma matches, distances, and punctuation, the classification is deterministic and requires no assessment of thematic, stylistic, or poetic similarity. We applied the same scheme across all sources: reusing the reference-type labels for the 270 references from Schropp et al. (2024b) and manually annotating the 275 newly added references and the 945 from Burns et al. (2021). Figure 3 describes the two positive reference types. Together with unrelated pairs, this yields a multiclass labeling into *no match*, *verbatim reference*, and *allusion*, enabling finer evaluation, since explicit lexical reuse, looser allusive correspondences, and pairs labeled *no match* pose substantially different retrieval and classification problems.

We release the complete dataset under the Apache 2.0 license as a Hugging Face collection comprising the source corpus, the query texts, and the annotated intertextual references.⁵ The fine-tuned retrieval and classification models reported in Section 5 are released as a companion Hugging Face collection.⁶

⁵<https://hf.co/collections/julian-schelb/datasets-for-latin-intertextuality-search>

⁶<https://hf.co/collections/julian-schelb/models-for-latin-intertextuality-search>

4 Experimental Setup

We use the dataset of Section 3 to answer three questions: (i) how well do retrieval-only methods rank true source segments among many candidates; (ii) how well do classifiers separate verbatim references, allusions, and pairs labeled *no match* on a fixed candidate set; and (iii) what precision–efficiency trade-off can a retrieve-and-rerank pipeline reach.

Task and pipeline. We define intertextuality detection as the identification of directional references between a *query set* (segments from chronologically later works) and a *source set* (segments from earlier works). A query segment can refer to one or more source segments, either as a verbatim reference or as an allusion; most query segments refer to none. Comparing 85k query against 91k source segments yields roughly 7.7 billion pairs, so exhaustive multiclass classification of every pair is computationally intractable. We therefore evaluate three method families: a retrieval-only approach that ranks candidates by similarity, a classification-only approach applied to a fixed candidate set, and a retrieve-and-rerank pipeline that classifies the top-*k* candidates produced by a retriever as *no match*, *verbatim reference*, or *allusion* (Figure 4).

4.1 Retrieval Models

We compare three retriever families. The sparse lexical methods TF-IDF and BM25 (Robertson and Zaragoza, 2009) rank source segments by term-weighted similarity and have no learned parameters; their terms are either surface tokens or lemmas produced by the Classical Language Toolkit (CLTK; Johnson et al., 2021), taken as unigrams or unigrams+bigrams. Two word-embedding (WE) baselines reimplement prior work at the word level: following Burns et al. (2021), the mean cosine similarity of bigram Word2Vec (Mikolov et al., 2013) vectors, and following Gong et al. (2025), position-aggregated LatinBERT (Bamman and Burns, 2020) token embeddings. The dense retrievers are bi-encoders fine-tuned with Online Contrastive Loss (Hadsell et al., 2006); we evaluate the multilingual encoders BGE-M3 (Chen et al., 2024), E5-small/base/large (Wang et al., 2024), and Granite-107m/278m (Awasthy et al., 2025), as well as the Latin-specific SPhilBERTa (Riemenschneider and Frank, 2023b). Following the E5 recipe, query and candidate are encoded with distinct `Query:` and `Candidate:` prefixes. Each

Model	Verbatim (cit.) Rec.			Allusion (cf.) Rec.		
	10	100	1000	10	100	1000
TF-IDF (surf. 1g)	60.63	75.39	87.55	8.52	23.59	50.97
TF-IDF (lem. 1g)	55.47	72.28	90.09	8.14	20.79	56.61
TF-IDF (lem. 1+2g)	58.19	74.03	90.78	7.90	25.70	60.74
BM25 (surf. 1g)	67.70	75.64	87.01	12.22	28.40	50.29
BM25 (lem. 1g)	63.97	77.62	90.08	11.96	29.43	57.92
BGE-M3	50.06	63.50	78.20	7.52	20.62	44.01
E5-large	49.93	63.06	79.54	7.34	18.25	48.42
E5-base	48.73	61.32	77.45	5.61	16.10	43.82
E5-small	44.17	58.03	74.63	5.68	13.99	34.56
Granite-278m	43.09	54.18	69.43	3.36	12.31	36.10
Granite-107m	31.35	41.04	57.56	2.27	6.68	27.28
SPhIBERTa	40.94	56.33	73.87	6.19	19.27	48.29
LatinBERT (WE)	56.70	68.84	82.83	10.03	28.14	52.70
Word2Vec (WE, 2g)	14.65	30.62	60.86	0.51	6.83	25.33

Table 2: **Performance comparison of retrieval approaches.** All values are $\times 100$, macro-averaged over query segments and averaged over the 5 folds. Lexical approaches (TF-IDF, BM25) use surface tokens (*surf*) or lemmas (*lem*) with unigrams (*1g*) or unigrams+bigrams (*1+2g*); WE denotes approaches based on word embeddings. Best value per column in bold.

word-embedding baseline uses the tokenizer and vocabulary released with its original model, which matters most for LatinBERT. Grouped model descriptions and training details are in Appendix D.

4.2 Classification Models

For classification we compare two families. As lexical reference points, we train a logistic regression (LogReg) and a histogram gradient-boosted decision tree (GBDT) on per-pair similarity features: TF-IDF cosine over lemma unigrams, lemma unigrams+bigrams, and character 3–4-grams, together with Jaccard overlap, raw-token overlap, and length features, all computed with the same CLTK pipeline as the sparse retrievers. Our main classifier is a cross-encoder that processes the query–candidate pair as a single concatenated input (Reimers and Gurevych, 2019), so self-attention can compare the two segments at the token level, which matters most for distinguishing allusions from verbatim references. We evaluate multilingual cross-encoders, namely mmBERT (Marone et al., 2026), XLM-RoBERTa (XLM-R; Conneau et al., 2020), ModernBERT (mBERT; Warner et al., 2025), and BERT-Romanian (Dumitrescu et al., 2020), together with the Latin-specific cross-encoders PhilBERTa and LaBERTa (Riemenschneider and Frank, 2023a), LatinBERT (Bamman and Burns, 2020), and RoBERTa-Latin (ClassCat, 2024). Table 3 reports the small/base or base/large variants where applicable. All cross-encoders are fine-tuned

Model	Verbatim (cit.)			Allusion (cf.)		
	P	R	F1	P	R	F1
LogReg (lexical)	37.26	83.31	46.00	2.47	36.95	4.45
GBDT (lexical)	35.56	84.28	44.55	2.16	22.46	3.76
mmBERT Base	75.35	75.44	73.67	8.54	44.66	12.67
mmBERT Small	71.94	77.01	71.86	8.90	37.08	12.79
XLM-R Large	75.44	81.07	75.65	6.68	53.23	10.87
XLM-R Base	68.54	74.57	68.71	6.77	55.83	10.91
mBERT Large	71.10	77.53	71.62	8.27	36.32	12.12
mBERT Base	68.73	74.64	69.10	8.07	41.61	12.03
LatinBERT	77.03	82.99	77.82	7.82	41.55	11.30
BERT-Romanian	62.10	70.52	63.29	7.26	42.14	11.33
PhilBERTa	61.75	72.98	63.65	2.80	45.25	5.04
LaBERTa	25.15	65.52	31.73	1.61	23.83	2.86
RoBERTa-Latin	0.81	26.48	1.53	1.02	11.54	1.74

Table 3: **Performance comparison of classification approaches.** All values are $\times 100$, macro-averaged over query segments and averaged over the 5 folds. Lexical approaches use sparse features; the remaining rows are fine-tuned transformer encoders. Decision thresholds are tuned per class and model on the training set. Best value per column in bold.

for 4 epochs with batch size 32 and learning rate 2×10^{-5} ; full feature definitions and training details are in Appendix E. The classifier is three-class throughout (*no match*, *verbatim reference*, *allusion*); for retrieval-style evaluation (Recall@ k and overall reference-detection F1) the two positive classes are merged into a single positive label, while reference-type-specific metrics are obtained by filtering the same predictions.

4.3 Metrics

Because most query–source pairs are labeled *no match*, accuracy is uninformative: a trivial all-negative predictor already scores near-perfectly. We therefore report per-class precision (P), recall (R), and F1 for verbatim references and allusions, and Recall@ k at $k \in \{10, 100, 1000\}$ for retrieval. Formal definitions and the global error-rate metrics (SMR, FPR, FNR) used in the appendix are given in Appendix B.

Aggregation. Per-class P, R, and F1 are computed per query segment (one-vs-rest, at each model’s tuned decision threshold), macro-averaged within each fold, and then averaged over the five folds (e.g., Table 3); Recall@ k is aggregated the same way.

4.4 Data Splits

The 1,490 references map onto 1,470 distinct query–source segment pairs, since 20 references share a pair with another reference. We use 5-fold cross-validation on these pairs, with all source segments

Fold	Queries		Sources		Pairs	
	Citing	+Distr.	Cited	+Distr.	Pos.	Neg.
0	142	+142	269	+269	277	152,515
1	141	+141	254	+254	270	142,986
2	141	+141	319	+319	331	179,585
3	141	+141	301	+301	312	169,452
4	141	+141	269	+269	280	151,436
Total	706	+706	1,412	+1,412	1,470	795,974
Mean	141.2	+141.2	282.4	+282.4	294.0	159,195

Table 4: **Per-fold evaluation set sizes** for classification and retrieve-and-rerank. Each fold contains query segments (*Citing*) that refer to at least one source segment (*Cited*); both are padded 1:1 with distractors from the same works (*+Distr.*). *Pos.* are distinct annotated segment pairs and *Neg.* all remaining query–source pairs.

of a given query assigned to the same fold; fold sizes therefore vary slightly. For classification and retrieve-and-rerank evaluation, each fold contains a query document (citing query segments plus an equal number of in-domain distractors, doubling the total) and a source document (cited source segments plus an equal number of segments that are not cited, drawn from the same source works, again doubling the total), and every query segment is scored against every source segment in the fold. Per-fold sizes are reported in Table 4: on average each fold spans ~ 282 queries and ~ 565 sources, yielding $\sim 159,000$ candidate pairs of which ~ 294 are ground-truth positives ($\sim 0.18\%$) and the remainder serve as in-domain negatives. Retrieval-only evaluation instead ranks the held-out queries against the complete source corpus of 90,546 segments, which is identical across folds. Consequently, retrieval cutoffs such as Recall@1000 remain meaningful and need not saturate.

4.5 Model Configuration and Ablation

Each training fold is augmented with negative pairs at positive-to-negative ratios from 1:1 to 1:10. For the dense retriever and the cross-encoder we compare base model families, negative-sampling ratios, learning rates, and epochs over the same 5 folds; sparse and WE methods need no fine-tuning and use default configurations. Per-class thresholds are tuned on the training fold with a plateau heuristic selecting the largest threshold within 0.01 of the best F1, which favors precision. Full grids are in Appendices D and E, with the corresponding results in Appendices G and H. Code and pipelines are released as the Python package `locisimiles` (Appendix K).

5 Experimental Results

We address the three questions of Section 4 in turn: retrieval-only ranking (Section 5.1), classification on a fixed candidate set (Section 5.2), and the precision–efficiency trade-off of the retrieve-and-rerank pipeline (Section 5.3).

5.1 Information Retrieval Results

Lexical methods match or outperform dense retrievers (Table 2): BM25 with lemmatized unigrams achieves the best Recall@100 on both reference types. Among dense retrievers, BGE-M3 and E5-large lead and SPhilBERTa trails the E5 family; LatInBERT word embeddings beat every bi-encoder, while static word vectors fall far behind. Allusion recall stays far below verbatim throughout. Full results and ablations on negative sampling and hyperparameters are in Appendix G.

Three factors help explain the advantage of lexical retrieval on allusions. First, lemmatization counters the fragmentation of Latin’s rich morphology under subword tokenization, which benefits BM25. Second, bi-encoders compress each segment into a single pooled embedding, which can obscure a short local cue in otherwise divergent context; the *nosce te* false negative in Section 6.2 illustrates this, and per-pair correctness correlates with lexical overlap for verbatim references but not for allusions (Section 6.1). Third, the benchmark retains a lexical component even among allusions, which share 1.65 lemmas on average; moreover, 275 of the 1,490 references were surfaced through n-gram candidate generation. As Section 5.2 shows, this disadvantage is specific to pooled bi-encoder retrieval: cross-encoders compare both passages at the token level and outperform lexical classifiers on allusions.

5.2 Classification Results

Transformer encoders clearly outperform the lexical baselines (Table 3). LogReg and GBDT achieve high verbatim recall but low precision, and near-zero precision on allusions. LatinBERT and XLM-R Large lead on verbatim, mmBERT Small on allusions. Allusions remain far harder than verbatim references throughout, limited by low precision despite moderate recall. Full results and ablations on negative sampling and hyperparameters are in Appendix H, with per-class breakdowns.

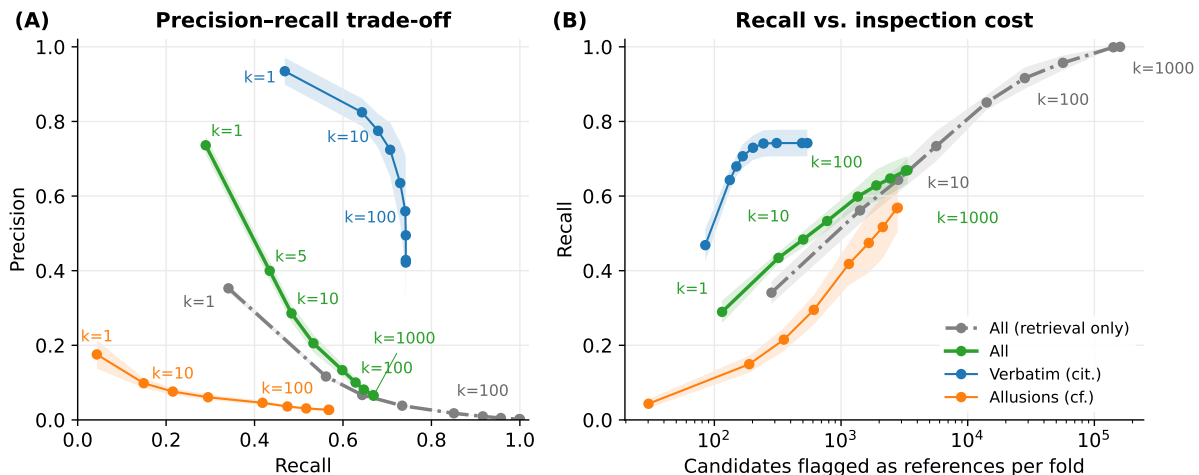


Figure 5: **Performance vs. efficiency trade-off in the retrieve-and-rerank pipeline.** The pipeline first retrieves the top- k candidates with BM25 over lemma unigrams, then applies a cross-encoder classifier to assign the labels *no match*, *verbatim reference*, or *allusion*. We compare this against a retrieval-only baseline where the top- k candidates are treated as positive predictions. In (B), the number of candidates flagged as references per fold corresponds to the passages a philologist would need to inspect. Selected k are labeled. Results are pooled per fold and averaged across 5 folds; shaded bands show ± 1 standard deviation.

5.3 Retrieve-and-Rerank Results

Guided by the per-stage findings, we instantiate the retrieve-and-rerank pipeline of Section 4 with BM25 over lemma unigrams as the first stage, the strongest retriever on Recall@100 in Section 5.1, and the cross-encoder classifier as the reranker, which Section 5.2 showed dominates the lexical baselines on precision. As a reference point we compare against a retrieval-only baseline that treats all top- k candidates as detected references.

The pipeline produces far fewer false positives than retrieval-only (Figure 5A). At $k = 5$ it reaches F1 0.42 against 0.19, and at $k = 100$ it still reaches 0.17 while retrieval-only falls to 0.02 (Table 12). In practical terms, at $k = 100$ the pipeline recovers 63% of true intertextual references (185 out of 294) while narrowing the candidate set from 28,240 to 1,901 passages, a 93% reduction in the space a philologist would need to inspect (Figure 5B; Appendix I).

6 Error Analysis

Complementing the aggregate results of Section 5, we now analyze where the XLM-R Large reranker succeeds and where it fails.

6.1 Quantitative Breakdown

The per-author-pair breakdown (Table 5) shows that classification quality depends largely on lexical overlap. For verbatim references the relationship is roughly monotonic: detection of high-overlap

pairs is close to the ceiling, while the low-overlap Valerius pairs are the weakest. The same factor explains the verbatim-allusion gap, since allusions share far fewer lemmas. Within the allusion class, however, overlap no longer predicts performance: Valerius allusions outperform Jerome’s despite lower overlap, suggesting that once overlap is scarce, the number of training pairs and source genre dominate. Valerius draws on well-attested poetic epic, whereas Jerome’s sparse allusions to prose sources such as Cicero are effectively missed. The few near-perfect verbatim scores ($F1 = 1.00$ for Jerome–Lucan, Jerome–Ovid, and Lactantius–Horace) rest on minimal support (≤ 5 pairs each) and should be read with caution. The breakdown separates two kinds of pair: those with enough shared lemmas to be matched, and those whose connection the surface text barely shows.

At the level of individual query–source pairs, the effect is strongly class-dependent (Figure 6). For verbatim references, lexical overlap predicts correctness strongly (Spearman $\rho = 0.36$ – 0.40 , $p < 0.001$); for allusions the correlation collapses to near zero and is not significant; and for pairs labeled *no match* it turns negative ($\rho \approx -0.09$), confirming that high overlap encourages spurious matches, the quantitative counterpart of the *omnia timeo* false positive below. The weakly negative overall correlation is therefore an artifact of class balance: pairs labeled *no match* dominate the pair grid, not evidence that overlap misleads in general.

Query	Source	Verbatim (cit.)			Allusion (cf.)		
		#	Ov.	F1	#	Ov.	F1
Jerome	Cicero	91	5.22	0.66	21	3.14	0.04
Jerome	Horace	41	6.80	0.98	—	—	—
Jerome	Lucan	2	5.00	1.00	—	—	—
Jerome	Ovid	2	6.00	1.00	—	—	—
Jerome	Virgil	211	6.95	0.89	38	3.20	0.12
Lactantius	Horace	5	12.40	1.00	—	—	—
Lactantius	Lucretius	24	13.21	0.99	—	—	—
Lactantius	Ovid	23	11.13	0.99	—	—	—
Lactantius	Propertius	3	8.00	1.00	—	—	—
Lactantius	Virgil	78	8.85	0.92	—	—	—
Valerius	Lucan	65	1.91	0.52	85	1.33	0.32
Valerius	Ovid	55	1.86	0.52	94	1.16	0.29
Valerius	Statius	42	2.26	0.49	78	1.70	0.23
Valerius	Virgil	207	2.13	0.53	305	1.57	0.14
Total		849	5.11	0.76	621	1.65	0.11

Table 5: **Per-author-pair performance breakdown for XLM-R Large.** Support (#), mean number of shared lemmas per pair (Ov.), and per-query macro F1 (averaged over 5 folds). Cells are shaded teal (darker = larger); “—” = no references of that type.

6.2 Qualitative Analysis

Two failure patterns recur. In a representative false positive, Jerome’s generic *omnia timeo* (“I fear everything”) overlaps with a Ciceronian passage, but his cue *ut ait gentilis poeta* (“as the pagan poet says”) in fact points to Virgil’s *omnia tuta timens*, whose distinctive word *tuta* is absent from the matched candidate: the model fires on a high-overlap distractor rather than the correct, lower-overlap target. In a representative false negative, the brief Socratic allusion *nosce te* (“know yourself”) opens a Jerome passage that then shifts to an unrelated discussion of stylistic simplicity, and the pooled sentence embedding loses the local cue. Both failure modes point to token- or span-level matching for short, low-overlap references. Full text of both examples, source pairs, and philological discussion are in Appendix J.1.

7 Discussion and Conclusion

In this work, we introduced *Loci Similes*, a benchmark dataset for Latin intertextuality detection that evaluates whether language models capture semantic similarity beyond exact lexical matching. Our experiments show that while the evaluated methods recover many long verbatim references and thematic allusions, detecting subtle short parallels, i.e., references that share only two or three words, remains difficult.

BM25 with lemmatized unigrams achieved the best retrieval performance (Recall@100 of 77.6

	Shared lemmas	Jaccard	Query coverage
No match n = 796k, 97% correct	−0.09	−0.07	−0.08
Verbatim (cit.) n = 849, 84% correct	0.36	0.40	0.37
Allusion (cf.) n = 621, 68% correct	0.04†	−0.01†	0.01†
All pairs n = 797k, 97% correct	−0.09	−0.07	−0.09

Spearman ρ

Figure 6: **Correlation between lexical overlap and per-pair correctness**, by gold class, for XLM-R Large on the full evaluation grid (Spearman ρ). All values significant at $p < 0.001$ except those marked †.

for verbatim references) and LatinBERT the best classification results (verbatim F1 of 77.8, Tables 2 and 3). Retrieving the top 100 candidates per query and reranking recovers 63% of true intertextual references while reducing the candidate set by 93%. The pipeline is a reference implementation rather than a production system (see Appendix L on scaling). Language models are therefore a promising avenue, but the central challenge, distinguishing meaningful reuse from coincidental lexical overlap, will require dedicated architectures.

Takeaways. First, a lemma-level BM25 baseline outperforms every dense retriever we evaluate: Latin’s rich morphology fragments under subword tokenizers, and CLTK lemmatization gives sparse methods an edge, motivating hybrid sparse–dense retrieval. The strongest neural retriever is not a pooled bi-encoder but token-level contextual embeddings, which nearly match the sparse retrievers on allusions. Second, neural cross-encoders show a clear advantage over lexical classifiers on allusions, with noticeably better precision on the subtler, low-overlap cases, while remaining competitive on verbatim references.

Future Work. Our error analysis suggests two directions: token-level matching (e.g., ColBERT (Khattab and Zaharia, 2020)) with objectives rewarding semantic over lexical correspondence, since pooled embeddings lose faint signals of short allusions; and span- or clause-level matching, since signals are often local yet embedded in divergent context. Future work could also characterize how an author uses an earlier text (appropriation vs. polemical engagement). Extending the benchmark to further citing authors would reduce its dependence on a few source genres.

Limitations

Data Coverage. Although our dataset comprises expert-verified positive pairs, it is not (and cannot be) exhaustive. The labeled dataset likely omits some valid references between the selected works. Consequently, it is possible that a few instances classified as “false positives” may represent genuine but undocumented intertextual references, potentially skewing the reported precision.

The 275 newly curated references were surfaced through n-gram candidate generation before philological verification. This procedure may favor references that retain some lexical overlap, so allusions without any shared lemma may be underrepresented within this subset. The remaining 1,215 references were adopted from existing scholarly datasets and did not pass through this candidate-selection step; the limitation therefore does not apply uniformly to the benchmark as a whole.

Operational Labels and Philological Ambiguity. Defining what constitutes a “reference” in classical philology remains a substantial methodological challenge. The boundaries between literal citation, subtle allusion, and general thematic resonance are fluid and subject to differing scholarly definitions. For a reproducible benchmark, however, annotators do not have to settle this question. Once an intertextual reference has been established, its benchmark type is assigned using the operational lemma-, proximity-, and punctuation-based criterion described in Section 3.3. While the taxonomy in Figure 3 illustrates broader philological interpretations, its subtypes do not alter the operational assignment. The resulting labels should therefore be understood as reproducible analytical categories rather than exhaustive interpretations of each intertext. Accordingly, the low F1 scores for allusions indicate that models struggle to predict this operational category; they do not, by themselves, demonstrate inconsistency or unreliability in the labels.

AI Usage Statement

Language model-based AI tools (Codex and GitHub Copilot) were used as coding assistants during implementation. Additionally, AI-based tools were used for grammar and spell checking. No AI-generated text appears in the final manuscript.

Acknowledgements

This work was supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation), project no. 382880410, *Quoting as a narrative strategy: A digital-hermeneutic analysis of intertextual references in Christian Late Antiquity*. We further thank the philologists whose expert annotation and verification made the benchmark possible, and the authors of the datasets we build on for making their resources publicly available.

References

- Parul Awasthy, Aashka Trivedi, Yulong Li, Mihaela A. Bornea, David Cox, Abraham Daniels, Martin Franz, Gabe Goodhart, Bhavani Iyer, Vishwajeet Kumar, Luis A. Lastras, J. Scott McCarley, Rudra Murthy, Vignesh P, Sara Rosenthal, Salim Roukos, Jaydeep Sen, Sukriti Sharma, Avirup Sil, Kate Soule, Md. Arafat Sultan, and Radu Florian. 2025. [Granite embedding models](#). *CoRR*, abs/2502.20204.
- David Bamman and Patrick J. Burns. 2020. [Latin BERT: A contextual language model for classical philology](#). *CoRR*, abs/2009.10053.
- Patrick J. Burns, James Brofos, Kyle Li, Pramit Chaudhuri, and Joseph P. Dexter. 2021. [Profiling of intertextuality in Latin literature using word embeddings](#). In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4900–4907.
- Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. [M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 2318–2335, Bangkok, Thailand.
- ClassCat. 2024. roberta-base-latin-v2: A RoBERTa base model for Latin. Hugging Face model card, <https://huggingface.co/ClassCat/roberta-base-latin-v2>. Trained on the Latin portion of CC-100.
- Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. [Unsupervised cross-lingual representation learning at scale](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451.
- Joseph P. Dexter, Pramit Chaudhuri, Patrick J. Burns, Elizabeth D. Adams, Thomas J. Bolt, Adriana Cásarez, Jeffrey H. Flynt, Kyle Li, James F. Patterson, Ariane Schwartz, and Scott Shumway. 2024. [A](#)

- database of intertexts in Valerius Flaccus' *Argonautica* 1: A benchmarking resource for the evaluation of computational intertextual search of Latin corpora. *Journal of Open Humanities Data*, 10(1).
- Arianna Di Bernardo, Simone Poetto, Pietro Sillano, Beatrice Villata, Weronika Sójka, Zofia Pietka-Danilewicz, and Piotr Pranke. 2021. [Latin writing styles analysis with machine learning: New approach to old questions](#). *CoRR*, abs/2109.00601.
- Stefan Daniel Dumitrescu, Andrei-Marius Avram, and Sampo Pyysalo. 2020. [The birth of Romanian BERT](#). In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4324–4328.
- Ashley Gong, Katy Gero, and Mark Schiefky. 2025. [Augmented close reading for classical Latin using BERT for intertextual exploration](#). In *Proceedings of the 5th International Conference on Natural Language Processing for Digital Humanities*, pages 403–417, Albuquerque, USA. Association for Computational Linguistics.
- Vanessa B. Gorman and Robert J. Gorman. 2016. [Approaching questions of text reuse in Ancient Greek using computational syntactic stylometry](#). *Open Linguistics*, 2(1):500–510.
- Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 1735–1742. IEEE.
- Maciej Janicki, Antti Kanner, and Eetu Mäkelä. 2023. [Detection and attribution of quotes in Finnish news media: BERT vs. rule-based approach](#). In *Proceedings of the 24th Nordic Conference on Computational Linguistics, NoDaLiDa 2023, Tórshavn, Faroe Islands, May 22-24, 2023*, pages 52–59. University of Tartu Library.
- Kyle P. Johnson, Patrick J. Burns, John Stewart, Todd Cook, Clément Besnier, and William J. B. Mattingly. 2021. [The Classical Language Toolkit: An NLP framework for pre-modern languages](#). In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 20–29.
- Omar Khattab and Matei Zaharia. 2020. [ColBERT: Efficient and effective passage search via contextualized late interaction over BERT](#). In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48. ACM.
- A. J. Kleywegt. 2005. *Valerius Flaccus, Argonautica, Book I*. Mnemosyne, Bibliotheca Classica Batava. Supplementum. Brill, Leiden.
- Iliia Kuznetsov, Jan Buchmann, Max Eichler, and Iryna Gurevych. 2022. [Revise and resubmit: An intertextual model of text-based collaboration in peer review](#). *Computational Linguistics*, 48(4):949–986.
- Pak Kin Lau and Stuart Michael McManus. 2024. [Mining asymmetric intertextuality](#). *CoRR*, abs/2410.15145.
- Ansel MacLaughlin, Shaobin Xu, and David A. Smith. 2021. [Recovering lexically and semantically reused texts](#). In *Proceedings of *SEM 2021: The Tenth Joint Conference on Lexical and Computational Semantics*, pages 52–66. Association for Computational Linguistics.
- Ananth Mahadevan, Michael Mathioudakis, Eetu Mäkelä, and Mikko Tolonen. 2025. [Text reuse in large historical corpora: insights from the optimization of a data science system](#). *International Journal of Data Science and Analytics*, 20(5):4631–4643.
- Enrique Manjavacas, Folger Karsdorp, and Mike Kestemont. 2020. [A statistical foray into contextual aspects of intertextuality](#). In *Proceedings of the Workshop on Computational Humanities Research (CHR 2020)*, volume 2723 of *CEUR Workshop Proceedings*, pages 77–96. CEUR-WS.org.
- Enrique Manjavacas, Brian Long, and Mike Kestemont. 2019. [On the feasibility of automated detection of allusive text reuse](#). In *Proceedings of the 3rd Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 104–114. Association for Computational Linguistics.
- Marc Marone, Orion Weller, William Fleshman, Eugene Yang, Dawn J. Lawrie, and Benjamin Van Durme. 2026. [mmBERT: A modern multilingual encoder with annealed language learning](#). In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*.
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. [Efficient estimation of word representations in vector space](#). In *1st International Conference on Learning Representations*.
- Hadar Miller, Tsvi Kuflik, and Moshe Lavee. 2025. [Text alignment in the service of text reuse detection](#). *Applied Sciences*, 15(6):3395.
- Fynn Petersen-Frey and Chris Biemann. 2024. [Fine-grained quotation detection and attribution in German news articles](#). In *Proceedings of the 20th Conference on Natural Language Processing*, pages 196–208. Association for Computational Linguistics.
- Nils Reimers and Iryna Gurevych. 2019. [Sentence-BERT: Sentence embeddings using Siamese BERT-networks](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3980–3990. Association for Computational Linguistics.

- Marie Revellio. 2022. *Zitate der »Aeneis« in den Briefen des Hieronymus: Eine digitale Intertextualitätsanalyse zur Untersuchung kultureller Transformationsprozesse*. De Gruyter, Berlin, Boston.
- Frederick Riemenschneider and Anette Frank. 2023a. Exploring large language models for classical philology. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15181–15199.
- Frederick Riemenschneider and Anette Frank. 2023b. *Graecia capta ferum victorem cepit. Detecting Latin allusions to Ancient Greek literature*. In *Proceedings of the Ancient Language Processing Workshop*, pages 30–38. Association for Computational Linguistics.
- Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389.
- David Rosson, Eetu Mäkelä, Ville Vaara, Ananth Mahadevan, Yann Ciarán Ryan, and Mikko Tolonen. 2023. Reception Reader: Exploring text reuse in early modern British publications. *Journal of Open Humanities Data*, 9(1):5.
- Franziska Schropp, Thomas E. Konrad, Marie Revellio, and Barbara Feichtinger. 2024a. Towards a differentiated digital-hermeneutic analysis tool for the detection of short quotations using the example of the Church Father Jerome. *Digital Humanities Quarterly*, 18(3).
- Franziska Schropp, Michael Wittweiler, Thomas E. Konrad, Marie Revellio, and Barbara Feichtinger. 2024b. Manuell-hermeneutische Forschung als Goldstandard: Zur Kalibrierung digitaler Analysetools in den Digital Humanities. *Digital Classics Online*, 10:76–98.
- Muhammad Sharjeel, Iqra Muneer, Sumaira Nosheen, Rao Muhammad Adeel Nawab, and Paul Rayson. 2023. Cross-lingual text reuse detection at document level for English–Urdu language pair. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 22(6):173:1–173:22.
- François Spaltenstein. 2002. *Commentaire des Argonautica de Valérius Flaccus (livres 1 et 2)*. Éditions Latomus, Bruxelles.
- Timothy R. Tangherlini and Ruofei Chen. 2024. Travels with BERT: Surfacing the intertextuality in Hans Christian Andersen’s travel writing and fairy tales through the network lens of large language model-based topic modeling. *Orbis Litterarum*, 79(6):519–562.
- Hawau Olamide Toyin, Federico Iezzi, Elia Scapini, Giulio Federico, and Giovanni Puccetti. 2026. Gretino: A Greek and Latin dataset to benchmark retrieval systems in classical languages. In *Proceedings of the Fifteenth Language Resources and Evaluation Conference (LREC 2026)*, pages 919–928, Palma, Mallorca, Spain. European Language Resources Association (ELRA).
- Tedo Vrbanec and Ana Mestrovic. 2023. Comparison study of unsupervised paraphrase detection: Deep learning - the key for semantic similarity detection. *Expert Systems*, 40(9).
- Jan Philip Wahle, Terry Ruas, Frederic Kirstein, and Bela Gipp. 2022. How large language models are transforming machine-paraphrase plagiarism. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 952–963. Association for Computational Linguistics.
- Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Multilingual E5 text embeddings: A technical report. *CoRR*, abs/2402.05672.
- Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Griffin Thomas Adams, Jeremy Howard, and Iacopo Poli. 2025. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2526–2547.
- Yuanchi Zhang and Yang Liu. 2022. DirectQuote: A dataset for direct quotation extraction and attribution in news articles. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 6959–6966. European Language Resources Association.
- Andrew Zissos. 2008. *Valerius Flaccus’ Argonautica Book I: Edited with Introduction, Translation, and Commentary*. Oxford University Press, Oxford.

A Corpus Sources

The corpus texts were aggregated from three digital repositories: *Corpus Corporum*⁷, the *Tesserae Project*⁸, and the *OpenGreekandLatin Project*. Table 6 lists the editions used; Figure 14 illustrates representative intertextual references from the ground truth.

B Evaluation Metrics

The classification model and the full retrieve-and-rerank pipeline are evaluated on the same directional comparison task between *query* and *source* documents, using 5-fold cross-validation over the 1,470 distinct positive segment pairs.

⁷<https://mlat.uzh.ch/>

⁸<https://github.com/tesserae/tesserae>

Author	Work	Source	Edition
Virgil	Aeneid	CC	Greenough (1900)
Virgil	Georgics	CC	Greenough (1900)
Virgil	Eclogues	CC	Greenough (1900)
Ovid	Amores	CC	Ehwald (1907)
Ovid	Ars Amatoria	CC	Ehwald (1907)
Ovid	Ex Ponto	CC	Wheeler (1939)
Ovid	Fasti	CC	Frazer (1933)
Ovid	Heroides	CC	Ehwald (1907)
Ovid	Ibis	CC	Merkel/Ehwald (1889)
Ovid	Medicamina	CC	Ehwald (1907)
Ovid	Metamorphoses	CC	Magnus (1892)
Ovid	Remedia Amoris	CC	Ehwald (1907)
Ovid	Tristia	CC	Wheeler (1939)
Martial	Epigrammata	CC	Heraeus (1925)
Lucretius	De Rerum Natura	CC	Martin (1934)
Lucan	Pharsalia	CC	Weise (1835)
Horace	Carmen Saeculare	CC	Shorey (1898)
Horace	Carmina	CC	Shorey (1919)
Horace	Ars Poetica	CC	Smart (1836)
Horace	Epistulae	CC	Fairclough (1929)
Horace	Epodes	CC	Vollmer (1912)
Horace	Saturae	CC	Smart (1836)
Catullus	Carmina	Tess	Merrill
Propertius	Elegiae	Tess	Mueller (1898)
Tibullus	Elegiae	Tess	Postgate (1915)
Cicero	<i>Opera Omnia</i>	Tess	<i>Varia</i>
Jerome	Epistulae	OGL	Hilberg (1910)
Jerome	<i>Varia</i>	CC	Patrologia Latina (1845)
Valerius Flaccus	Argonautica	CC	Kramer (1913)
Statius	Thebais	CC	Mozley (1928)

Table 6: **Corpus data sources.** Abbreviations: CC (Corpus Corporum), Tess (Tesserae Project), and OGL (OpenGreekandLatin Project).

Unlike standard information retrieval tasks that focus on ranking top- k candidates, our objective is to classify the full set of possible references between the query and source documents. Because many query segments have no true positive match in the source document, evaluation must reward the correct rejection of pairs labeled *no match* as well as the recovery of attested references. We therefore define $N = TP + FP + FN + TN$ as the total number of text-segment pairs and use the following error-based metrics, which better match the practical constraints of philological workflows.

- **Segment-Misclassification Rate (SMR):** defined as the fraction of all query-source pairs that were misclassified. This serves as a global error rate. Values range from 0 (perfect retrieval) to 1 (complete failure).

$$SMR = \frac{FP + FN}{N}$$

- **Global False-Positive Rate (FPR):** defined as the share of the total dataset incorrectly predicted as references. A high FPR indicates a system

prone to “over-generating” candidate references.

$$FPR = \frac{FP}{N}$$

- **Global False-Negative Rate (FNR):** defined as the share of the total dataset that contains true references missed by the system. A high FNR indicates that genuine intertextual references remain undiscovered.

$$FNR = \frac{FN}{N}$$

Together, these metrics decompose the total error (SMR) into false-positive and false-negative components, making it clear whether a model tends to over-generate candidate references or miss genuine references. We calculate each metric per query segment and report the mean across queries.

C Data Split and Evaluation Setup

All retrieval, classification, and retrieve-and-rerank experiments use the same 5-fold split over the 1,470 distinct positive segment pairs. The split is stratified at the query level: all positive targets of a given query segment are assigned to the same fold, so no query appears in both training and held-out evaluation. Fold sizes therefore vary slightly, because individual queries can cite different numbers of source segments, even though the number of query segments per fold is approximately balanced.

Document-level, fully pairwise evaluation. For classification and retrieve-and-rerank evaluation, each fold uses a reduced pairwise document pool. The *query document* contains held-out query segments plus an equal number of query-side distractors drawn from the same work but without annotated references; the *source document* contains the ground-truth source segments plus an equal number of source-side distractors drawn from the cited works. We then score *every* query segment against *every* source segment in that fold. The evaluation is therefore fully pairwise rather than restricted to gold-aligned pairs, forcing models to distinguish true references from in-domain distractors written by the same authors and in similar registers. Retrieval-only experiments use the same held-out query split but rank each query against the complete source corpus of 90,546 segments, which is identical in every fold.

Per-fold sizes. Per-fold sizes (queries, sources, ground-truth pairs, and total pairwise comparisons) are reported in Table 4 in Section 4.4.

The reduced per-fold query and source documents are reused across classification models and retrieve-and-rerank runs; all retrieval-only models use the separate full-corpus pool described above. Each fold then requires roughly 1.6×10^5 pairwise scores, of which only $\sim 0.18\%$ are true positives; the global error-rate metrics in Appendix B make this imbalance visible.

D Retrieval Setup

This appendix covers the retrieval models, ablations, negative sampling, and implementation; results are in Appendix G.

D.1 Model Families

We group the retrieval methods by the kind of signal they exploit, from sparse lexical overlap to fine-tuned dense bi-encoders.

Sparse retrievers. We evaluate TF-IDF and BM25 over both surface tokens and CLTK lemmas, using either unigrams alone or unigrams plus bigrams. For each query segment, source segments are ranked by sparse vector similarity (TF-IDF) or BM25 score, without learned parameters. TF-IDF uses a custom analyzer over the CLTK-tokenized input with `sublinear_tf = True`, `max_features = 50k`, `min_df = 1`, and `max_df = 1.0`, fit on the training fold only.

Word-embedding retrievers. We reimplement two prior approaches that operate on word-level rather than sentence-level embeddings. Following Burns et al. (2021), we score segment pairs by the mean cosine similarity of their bigram Word2Vec vectors, testing a Bamman–Burns lemma Word2Vec checkpoint⁹ and a LiLa fastText skip-gram checkpoint.¹⁰ Following Gong et al. (2025), we use position-aggregated contextual LatinBERT¹¹ token embeddings as the signal.

⁹`qcl_bamman_lemma_300`

¹⁰`lila_fasttext_skip_win5_min5`

¹¹`ashleygong03/bamman-burns-latin-bert`, tokenized with the subword encoder released with the original model; a WordPiece tokenizer over its converted vocabulary instead maps about 60% of Latin words to [UNK]. The same encoder is used for the fine-tuned LatinBERT cross-encoder of Table 3. CLTK lemmas are mapped to the u/i orthography of the static vocabularies before lookup.

Dense retrievers. For sentence-level embedding, we evaluate the multilingual E5 family (Small, Base, and Large) (Wang et al., 2024), the Granite embedding models (107m and 278m) (Awasthy et al., 2025), and BGE-M3 (Chen et al., 2024). We also include SPhilBERTa (Riemenschneider and Frank, 2023b), a sentence-transformer model for Ancient Greek, Latin, and English built on the PhilBERTa encoder (Riemenschneider and Frank, 2023a). All dense retrievers are used as bi-encoders that embed queries and sources independently and rank candidates by cosine similarity.

D.2 Ablation Design

We evaluate model choice and training regime for the retrieval stage independently before combining the selected components in the retrieve-and-rerank pipeline.

- **Base-model ablations.** We compare sparse lexical retrievers, word-embedding baselines, and fine-tuned bi-encoders to isolate the effect of lexical matching, word-level semantics, and sentence-level dense retrieval.
- **Negative-sampling ablations.** For the learned bi-encoder, we compare multiple strategies for constructing negatives and vary the positive-to-negative training ratio over the grid 1:1, 1:2, $\dots$, 1:10 (ten ratios) to measure sensitivity to class imbalance.
- **Hyperparameter ablations.** For the fine-tuned bi-encoder, we vary the learning rate $\in \{1 \times 10^{-5}, 2 \times 10^{-5}, 3 \times 10^{-5}\}$ and number of epochs $\in \{1, 2, 4, 6\}$ (12 configurations $\times$ 5 folds = 60 runs) to identify stable operating regions rather than relying on a single default setting.
- **Pipeline-depth ablations.** For the full retrieve-and-rerank system, we vary retrieval depth over 15 values, from $k = 1$ to $k = 1000$ (1, 5, 10, 20, 50, and then 100–1000 in steps of 100), to measure the trade-off between recall, false positives, and the number of candidate pairs inspected or reranked. A retrieval-only baseline treats the top- k candidates as positive predictions; the reranker is applied only to this pool (results in Appendix I).

D.3 Negative Sampling

The quality of the learned bi-encoder depends on the negative examples used during training. We

evaluated four negative sampling strategies:

- **Random pairs** ($\langle \text{random}, \text{random} \rangle$): Pairs are formed by selecting two completely disjoint segments at random from the corpus.
- **Random negatives** ($\langle \text{query}, \text{random} \rangle$): For each positive query, we sample a negative candidate uniformly at random from the remaining corpus.
- **Hard negatives** ($\langle \text{query}, \text{similar} \rangle$): A pre-trained embedding model identifies “hard negatives”: candidates that are semantically similar to the query but are not intertextual references.
- **Mixed negatives** ($\langle \text{query}, \text{mix} \rangle$): Uniformly sampled negatives and semantically similar hard negatives are combined for the same query.

D.4 Implementation Details

We evaluate the sparse retrievers (TF-IDF, BM25) over both surface tokens and CLTK lemmas, with unigrams alone and unigrams plus bigrams. All vectorizers share the CLTK tokenization and lemmatization pipeline and are fit on the training fold only.

For the dense retriever, we employ sentence-transformer bi-encoders that generate embeddings for query and source segments independently. During inference, the bi-encoder computes cosine similarity between the query embedding and every source embedding in the index, and candidates are ranked by this score. To adapt the retriever to the task, we fine-tune the model using Online Contrastive Loss, which pulls linked pairs together and pushes unrelated pairs apart.

We fine-tuned the bi-encoder for 4 epochs (batch size 32, learning rate 2×10^{-5} , AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, `max_grad_norm` = 1.0, weight decay 0.01) using a linear learning-rate scheduler with 10% warmup and, following the E5 recipe (Wang et al., 2024), prepended “Query: ” to query segments and “Candidate: ” to source segments before encoding. Evaluation during training uses batch size 8 and is run every 50 optimizer steps; we use a fixed seed of 42 across folds. The bigram Word2Vec retriever uses a public checkpoint trained on the Bamman–Burns Latin corpus.

E Classification Setup

The multiclass reranking setup covers model families, ablation design, sampling strategies, and implementation details. The reranker assigns one of three labels, 0 = *no match*, 1 = *cit.*, or 2 = *cf.*, to each (query, candidate) pair. Classification results are reported in Appendix H.

E.1 Model Families

We group the classifiers by the kind of signal they exploit.

Sparse classifiers. As lexical reference points for reranking, we train a logistic regression (LogReg) and a histogram gradient-boosted decision tree (GBDT) on pairwise similarity features computed for each (query, candidate) pair. These models test how far explicit lexical evidence alone can separate *no match*, *verbatim reference*, and *allusion*.

Neural cross-encoders. For reranking, we evaluate multilingual and domain-adapted transformer encoders as multiclass cross-encoders. The multilingual baselines include XLM-R (Base and Large) (Conneau et al., 2020), ModernBERT (Base and Large) (Warner et al., 2025), and mmBERT (Small and Base) (Marone et al., 2026). To measure the impact of domain adaptation, we evaluate PhilBERTa and LaBERTa (Riemenschneider and Frank, 2023a), LatinBERT (Bamman and Burns, 2020), and RoBERTa-Latin (ClassCat, 2024), which are pre-trained on Latin corpora. We also include BERT-Romanian (Dumitrescu et al., 2020) to test cross-lingual transfer from a related Romance language. Unlike the bi-encoder retrievers, cross-encoders process the query and candidate jointly and predict one of the three relation labels directly.

E.2 Ablation Design

We evaluate model choice and training regime for the reranking stage independently before combining the selected components in the retrieve-and-rerank pipeline.

- **Base-model ablations.** We compare sparse lexical classifiers against multilingual and domain-adapted transformer cross-encoders.
- **Negative-sampling ablations.** We compare multiple strategies for constructing negatives and vary the positive-to-negative training ratio

over the grid 1:1, 1:2, . . . , 1:10 (ten ratios) to measure sensitivity to class imbalance. Pair labels are looked up in a precomputed (query, corpus) table; when a pair carries both a *cit.* and a *cf.* annotation, *cit.* takes precedence.

- **Three-way class-sampling ablations.** In a follow-up experiment for the 3-class classifier, we vary the target sampling proportions of the three training labels, 0 = *no match*, 1 = *cit.*, and 2 = *cf.*, using a target-class-probability sampler. The default rerun uses a mild fixed exposure of {0:0.80, 1:0.10, 2:0.10}; the ablation also tests {0.50, 0.25, 0.25}, {0.60, 0.20, 0.20}, {0.33, 0.33, 0.33}, and a “natural” setting without resampling. This tests how strongly the reranker depends on the relative frequency of negatives, verbatim references, and allusions during training.

E.3 Negative Sampling and Class Sampling

The quality of the multiclass reranker depends on the negative examples and class-mixing strategy used during training. We evaluated four negative sampling strategies:

- **Random pairs** (`<random, random>`): Pairs are formed by selecting two completely disjoint segments at random from the corpus.
- **Random negatives** (`<query, random>`): For each positive query, we sample a negative candidate uniformly at random from the remaining corpus.
- **Hard negatives** (`<query, similar>`): A pre-trained embedding model identifies “hard negatives”: candidates that are semantically similar to the query but are not intertextual references.
- **Mixed negatives** (`<query, mix>`): Uniformly sampled negatives and semantically similar hard negatives are combined for the same query.

In addition to choosing how negatives are constructed, the multiclass reranker requires a decision about how often each of the three labels is shown during training. After remapping the labels to 0 = *no match*, 1 = *cit.*, and 2 = *cf.*, training batches are drawn with a target-class-probability sampler rather than relying only on the raw empirical class frequencies. The default setting uses class

probabilities {0:0.80, 1:0.10, 2:0.10}, and the ablation varies these three-way proportions to measure the effect of emphasizing negatives versus the two positive reference types during training.

E.4 Implementation Details

The sparse classifiers (logistic regression and GBDT) use pairwise features for each (query, candidate) pair: TF-IDF cosine similarities over lemma unigrams, lemma unigrams+bigrams, and character 3–4-grams, plus Jaccard overlap on lemmas, raw token overlap, and length features. All vectorizers share the retrievers’ CLTK tokenization and lemmatization pipeline and are fit on the training fold only.

For the cross-encoder, we concatenate query and candidate, separated by the model’s special tokens (Figure 4), and truncate each to 50% of the token budget to prevent either from dominating.

`<s> Legimus ... </s></s> excudent... </s>`

We fine-tuned the cross-encoder for 4 epochs (per-GPU batch size 32 for base models and 16 for large models, with gradient accumulation chosen to keep an effective batch size of 64 across GPUs, learning rate 2×10^{-5} , AdamW) to predict the multiclass relation label for each query-candidate pair and minimize cross-entropy against the ground-truth labels. RoBERTa-family encoders use an explicit separator `fix` to insert the correct `</s></s>` sequence between query and candidate.

In the 3-class classifier follow-up, the cross-encoder uses a target-class-probability sampler during training. This sampler is implemented as a `WeightedRandomSampler` that is re-drawn each epoch, so that the empirical class exposure matches the configured target within each fold. We do not additionally re-weight the cross-entropy loss, so the sampler is the only mechanism used for the sampling-proportion ablation described above.

For reporting, per-class decision thresholds are tuned on the training fold using a *plateau-high* heuristic with tolerance 0.01: starting from the highest threshold, we walk down the precision–recall curve and pick the largest threshold at the top of the F1 plateau, independently for the *cit.* and *cf.* classes. The remaining mass is assigned to *no match*. These tuned thresholds are then frozen and applied to the held-out fold.

F Retrieve-and-Rerank Setup

This experiment evaluates the full retrieve-and-rerank system on the fold-level evaluation sets described in Appendix C. Unlike retrieval-only evaluation, this experiment uses the reduced fold-level source pool. For each held-out fold, the retriever first scores every query segment against the source document and returns the top- k candidates; the reranker then classifies only this candidate pool. All remaining query–source pairs are treated as negative predictions. This preserves the fully pairwise document-level evaluation while measuring how much of the exhaustive comparison grid can be avoided through candidate generation.

The final retrieve-and-rerank configuration uses BM25 with lemma unigrams (the strongest sparse retriever; see Appendix G) as the candidate generator. The reranking stage uses the fine-tuned XLM-R Large cross-encoder; the pipeline was not re-run with LatinBERT, whose verbatim F1 is slightly higher. For each candidate pair, the reranker predicts 0 = *no match*, 1 = *cit.*, or 2 = *cf.*; the two positive classes are merged when reporting reference-detection performance, and reference-type-specific metrics are computed by filtering the same cached predictions.

We compare this retrieve-and-rerank system against a retrieval-only baseline at the same retrieval depths. The retrieval-only baseline assigns all top- k candidates to the positive class and all other pairs to *no match*; the retrieve-and-rerank pipeline instead lets the classifier decide which retrieved candidates remain positive. Retrieval depth is swept over 15 values, from $k = 1$ to $k = 1000$ (1, 5, 10, 20, 50, and then 100–1000 in steps of 100); the reported retrieve-and-rerank results in Appendix I focus on representative depths from this grid.

G Retrieval Results

We evaluate candidate retrieval with standard information-retrieval metrics, focusing on whether the first stage preserves enough true references for downstream reranking. Table 7 and Figure 7 show that the larger multilingual embedding models have the strongest recall among the dense retrievers. Table 8 reports the lexical and word-embedding retrieval baselines. Figure 8 gives the fold-level behavior of the best-performing model, while Figure 10 and Figure 11 summarize the negative-ratio and hyperparameter ablations.

H Classification Results

We evaluate the multiclass cross-encoder with standard classification metrics and the task-specific global error rates defined in Appendix B.

H.1 Sampling and Models

Table 9 shows that query-conditioned random negatives ($\langle \text{qry}, \text{rnd} \rangle$) reduce global error rates more effectively than fully random pairs ($\langle \text{rnd}, \text{rnd} \rangle$). Table 10 reports the 3-class model under different class-sampling proportions. Table 11 compares the base architectures, while Figure 9 summarizes the negative-ratio sweep.

Three patterns stand out. First, the composition of the negatives matters more than their number: mixed negatives ($\langle \text{qry}, \text{mix} \rangle$) yield the smallest global error (SMR 0.045) and the fewest false positives (7,084), query-conditioned random negatives come next (SMR 0.063), whereas fully random pairs let the false positives explode to 44,578 (SMR 0.289) because the model never sees a semantically plausible distractor during training; hard negatives alone (SMR 0.111) also fall short of the mixture, so easy negatives remain necessary to calibrate the decision boundary. Second, raising the share of negatives from 1:1 to 1:10 lowers the false-positive rate monotonically (FPR 0.19 to 0.03) at a modest cost in recall (0.67 to 0.61; Table 10), which is the trade-off behind the 1:10 default. Third, among the encoders (Table 11) LatinBERT reaches the highest verbatim F1 (0.78) and macro F1 (0.45), within a fold std of XLM-R Large (0.76, 0.43) and mmBERT Base (0.74, 0.43); mmBERT Small has the best allusion F1 (0.13), BERT-Romanian (0.63) matches PhilBERTa (0.64), and LaBERTa (0.32) and RoBERTa-Latin (0.02) fall behind, and the lexical classifiers combine the highest verbatim recall (GBDT 0.84) with precision below 0.4.

I Retrieve-and-Rerank Results

Table 12 compares the retrieval-only baseline with the retrieve-and-rerank pipeline at varying retrieval depths (k), relating performance gains to the number of candidate pairs sent to the classifier. Figure 12 visualizes the same precision–efficiency trade-off.

Two observations follow from Table 12. At $k = 5$ the reranker keeps 321 of the 1,412 retrieved candidates, retaining 127 of the 164 true references in the pool while cutting the false positives from 1,248 to 193 (FPR $\times 10^4$ from 39.4 to 6.1). As k

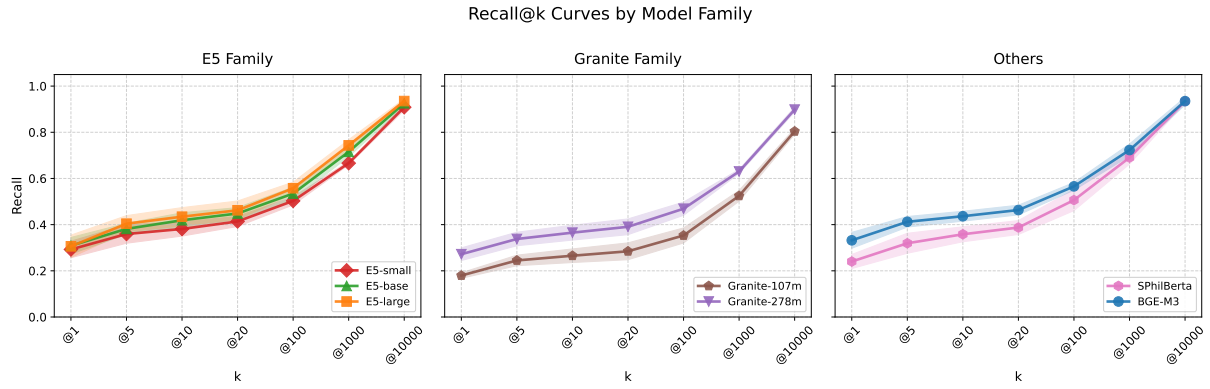


Figure 7: **Recall@k comparison across embedding model families.** Performance of E5, Granite, BGE-M3, and the domain-specific SPhILBERTa on the Latin intertextuality retrieval task. We observe that larger models show better performance, with multilingual E5-large achieving the largest scores overall. Error bands show ± 1 std across 5-fold cross-validation.

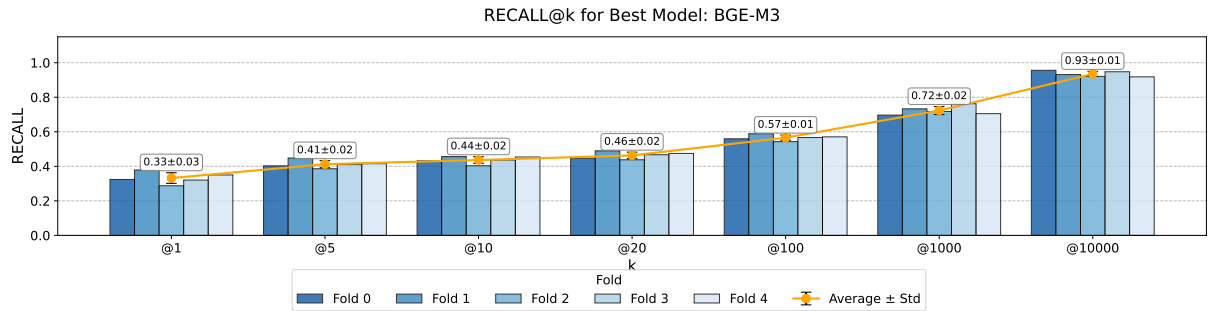


Figure 8: **Recall@k performance of the best embedding model across individual folds.** We show recall at varying cutoff values ($k \in \{1, 5, 10, 20, 100, 1000, 10000\}$) for the top-performing model on the Latin intertextuality retrieval task. Each bar represents a single fold from 5-fold cross-validation, with the orange line indicating mean $\pm$ std. Performance improves substantially with larger k, approaching near-perfect recall at k = 10000.

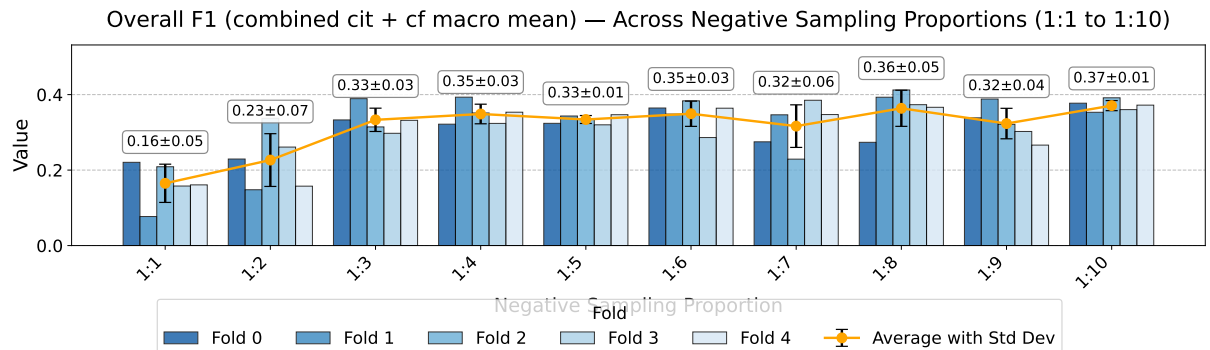


Figure 9: **Impact of training data imbalance on classification performance.** The plot illustrates the overall F1 (macro mean over cit. and cf.) on the evaluation set as the positive-to-negative training ratio is increased from 1:1 to 1:10. Shaded areas represent the standard deviation across the 5 cross-validation folds.

Model	Recall			MRR		
	@10	@100	@1000	@10	@100	@1000
BGE-M3	0.436 ± 0.021	0.566 ± 0.017	0.723 ± 0.026	0.402 ± 0.026	0.408 ± 0.025	0.408 ± 0.025
E5-large	0.434 ± 0.039	0.558 ± 0.026	0.743 ± 0.024	0.384 ± 0.043	0.391 ± 0.043	0.391 ± 0.043
E5-base	0.419 ± 0.032	0.535 ± 0.028	0.715 ± 0.021	0.377 ± 0.033	0.383 ± 0.032	0.384 ± 0.032
E5-small	0.381 ± 0.031	0.503 ± 0.017	0.666 ± 0.007	0.356 ± 0.029	0.362 ± 0.028	0.362 ± 0.028
Granite-278m	0.365 ± 0.032	0.469 ± 0.029	0.630 ± 0.009	0.327 ± 0.018	0.333 ± 0.017	0.333 ± 0.017
Granite-107m	0.265 ± 0.029	0.353 ± 0.033	0.525 ± 0.023	0.229 ± 0.009	0.233 ± 0.009	0.234 ± 0.009
SPhilBERTa	0.358 ± 0.032	0.506 ± 0.046	0.690 ± 0.028	0.303 ± 0.031	0.310 ± 0.030	0.311 ± 0.030

Table 7: **Performance comparison of different base models for the dense retrieval task.** Each held-out query is ranked against the full source corpus of 90,546 segments; Recall@ k and MRR@ k are computed per query with both reference types pooled (mean $\pm$ std over 5 folds). All bi-encoders were fine-tuned with the same negative-sampling strategy and training schedule (Online Contrastive Loss, 4 epochs, learning rate 2×10^{-5} ; Appendix D). The larger multilingual encoders (BGE-M3, E5-large) lead on every metric; Table 2 reports the same runs split by reference type. Best value per column in bold.

Model	Recall			MRR		
	@10	@100	@1000	@10	@100	@1000
TF-IDF (surf. 1g)	0.516 ± 0.037	0.652 ± 0.024	0.804 ± 0.019	0.439 ± 0.035	0.446 ± 0.034	0.446 ± 0.034
TF-IDF (lem. 1g)	0.475 ± 0.050	0.623 ± 0.027	0.821 ± 0.014	0.431 ± 0.048	0.438 ± 0.047	0.439 ± 0.046
TF-IDF (lem. 1+2g)	0.494 ± 0.039	0.645 ± 0.040	0.833 ± 0.021	0.419 ± 0.037	0.426 ± 0.037	0.426 ± 0.036
BM25 (surf. 1g)	0.572 ± 0.041	0.666 ± 0.020	0.799 ± 0.020	0.556 ± 0.032	0.560 ± 0.031	0.561 ± 0.031
BM25 (lem. 1g)	0.545 ± 0.046	0.680 ± 0.011	0.824 ± 0.007	0.514 ± 0.056	0.521 ± 0.053	0.522 ± 0.053
Word2Vec (WE, 2g)	0.122 ± 0.012	0.264 ± 0.023	0.529 ± 0.033	0.077 ± 0.005	0.083 ± 0.006	0.084 ± 0.006
fastText (WE, LiLa)	0.131 ± 0.017	0.280 ± 0.042	0.517 ± 0.038	0.073 ± 0.015	0.079 ± 0.014	0.080 ± 0.014
LatinBERT (WE)	0.485 ± 0.025	0.616 ± 0.022	0.765 ± 0.009	0.452 ± 0.032	0.458 ± 0.032	0.459 ± 0.032

Table 8: **Performance comparison of baseline retrieval approaches.** Same protocol as Table 7: full source corpus, both reference types pooled, mean $\pm$ std over 5 folds. Lexical approaches (TF-IDF, BM25) use surface tokens (*surf.*) or CLTK lemmas (*lem.*) with unigrams (*1g*) or unigrams+bigrams (*1+2g*) and have no learned parameters. WE denotes word-embedding baselines: bigram Word2Vec vectors following Burns et al. (2021), a LiLa fastText checkpoint, and position-aggregated LatinBERT token embeddings following Gong et al. (2025) (tokenization: Appendix D). BM25 over lemma unigrams gives the best Recall@100 and is the first stage of the pipeline; the LatinBERT embeddings are the strongest neural retriever, ahead of every bi-encoder in Table 7. Best value per column in bold.

Sampling Method	Classification Metrics				Global Error Rates			Confusion Matrix			
	Prec.	Rec.	F1	Acc.	FPR	FNR	SMR	TP	FP	FN	TN
Mixed negatives ⟨qry, mix⟩	0.03	0.61	0.05	0.96	0.0443	0.0007	0.0450	179	7084	114	152150
Random pairs ⟨rnd, rnd⟩	0.01	0.67	0.01	0.71	0.2888	0.0006	0.2894	195	44578	99	114661
Random negatives ⟨qry, rnd⟩	0.02	0.66	0.04	0.94	0.0626	0.0006	0.0633	193	10154	101	149085
Hard negatives ⟨qry, sim⟩	0.01	0.61	0.02	0.89	0.1100	0.0007	0.1107	179	17554	114	141682

Table 9: **Performance comparison of the classification model across different negative sampling strategies.** Metrics are computed from the thresholded multiclass evaluation. The 3-class confusion matrix is collapsed over the two positive classes (*cit.* and *cf.*); wrong positive-type predictions count as class-level errors. FPR, FNR, and SMR denote global error rates normalized by the total number of pairs N . Results are averaged across 5 folds. Best value per column in bold.

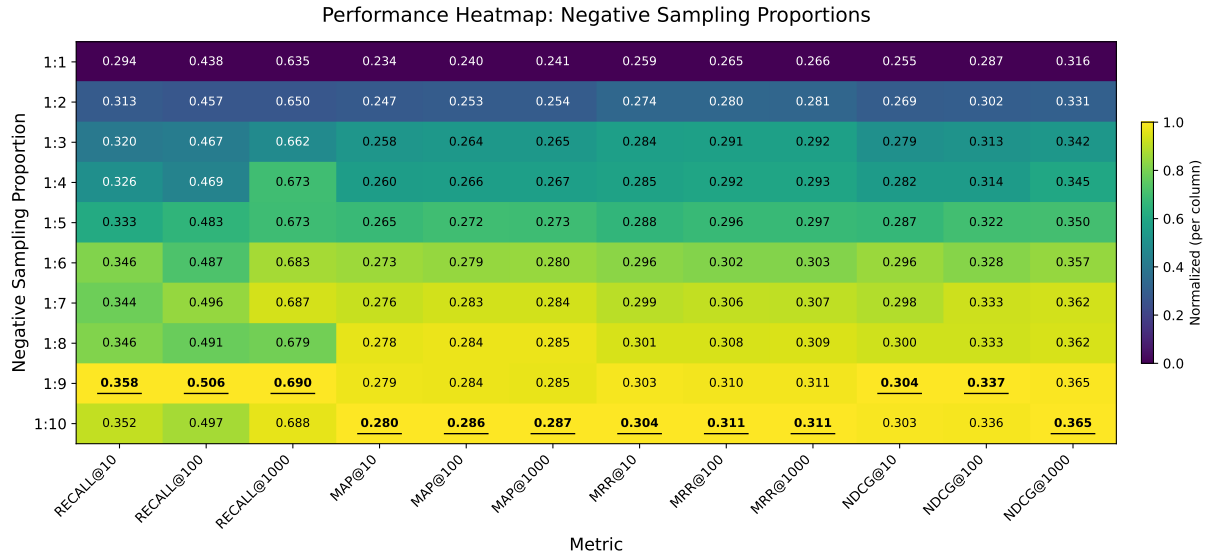


Figure 10: **Impact of training data imbalance on retrieval.** Heatmap showing recall, MAP, MRR, and NDCG scores at different cutoff values (@10, @100, @1000) across negative sampling proportions (1:1 to 1:10). Values are averaged over 5 cross-validation folds. Colors are normalized per column to highlight relative performance differences. Best values per metric are shown in bold with an orange underline. Larger ratios of negative samples generally improve retrieval performance, with optimal results often achieved at ratios of 1:5 or larger.

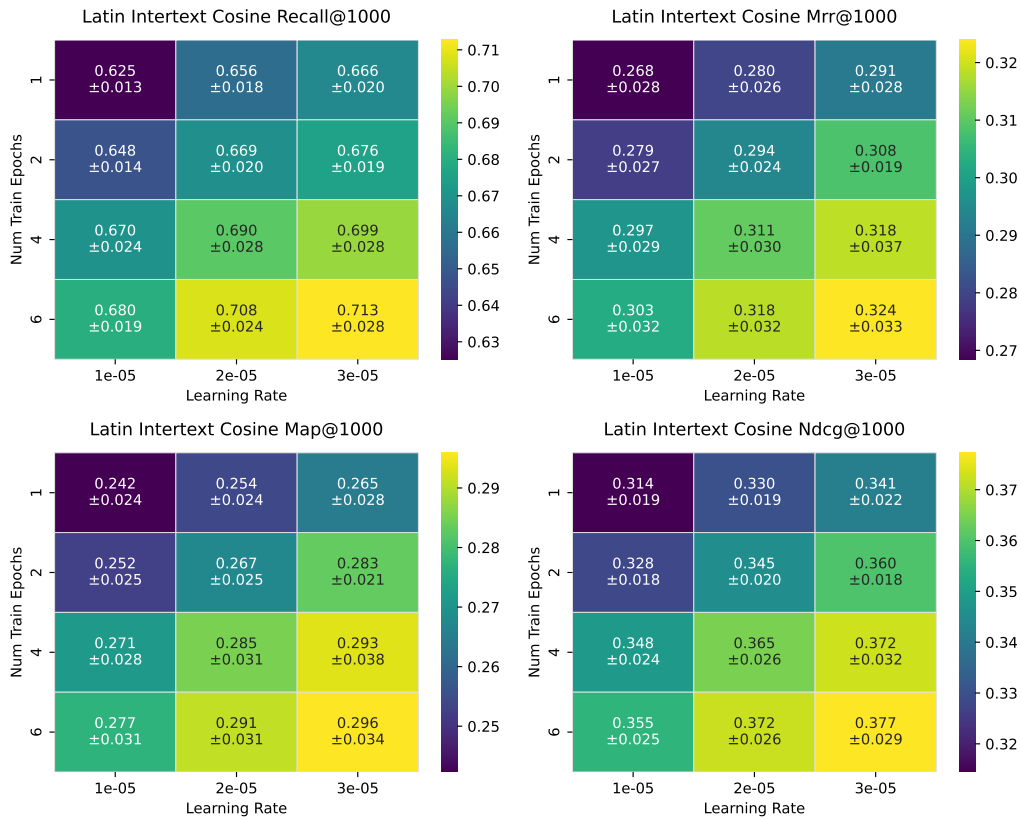


Figure 11: **Hyperparameter sensitivity for the dense retriever.** We visualize retrieval metrics (recall, MRR, MAP, and NDCG at $k = 1000$) across varying learning rates and training epochs. Results are averaged across 5 folds, where annotated values represent the mean score $\pm$ standard deviation.

Sampling Proportion	Classification Metrics				Global Error Rates			Confusion Matrix			
	Prec.	Rec.	F1	Acc.	FPR	FNR	SMR	TP	FP	FN	TN
1 : 1 neg. {0: 0.50, 1: 0.29, 2: 0.21}	0.01	0.67	0.02	0.81	0.1917	0.0006	0.1924	195	30148	98	129099
1 : 2 neg. {0: 0.67, 1: 0.19, 2: 0.14}	0.01	0.69	0.02	0.83	0.1704	0.0006	0.1709	202	27353	91	131896
1 : 3 neg. {0: 0.75, 1: 0.14, 2: 0.11}	0.03	0.64	0.07	0.96	0.0401	0.0007	0.0407	188	6340	106	152910
1 : 4 neg. {0: 0.80, 1: 0.12, 2: 0.08}	0.03	0.70	0.05	0.94	0.0640	0.0006	0.0646	205	10643	88	148601
1 : 5 neg. {0: 0.83, 1: 0.10, 2: 0.07}	0.02	0.68	0.03	0.91	0.0924	0.0006	0.0930	200	14984	93	144253
1 : 6 neg. {0: 0.86, 1: 0.08, 2: 0.06}	0.03	0.65	0.06	0.96	0.0396	0.0006	0.0402	191	6177	102	153063
1 : 7 neg. {0: 0.88, 1: 0.07, 2: 0.05}	0.02	0.66	0.04	0.94	0.0590	0.0006	0.0596	194	9520	100	149714
1 : 8 neg. {0: 0.89, 1: 0.06, 2: 0.05}	0.04	0.66	0.07	0.96	0.0373	0.0006	0.0380	193	5894	101	153345
1 : 9 neg. {0: 0.90, 1: 0.06, 2: 0.04}	0.03	0.63	0.06	0.95	0.0480	0.0007	0.0487	184	7491	109	151743
1 : 10 neg. {0: 0.91, 1: 0.05, 2: 0.04}	0.04	0.61	0.08	0.97	0.0335	0.0007	0.0342	179	5245	115	153993

Table 10: **Performance comparison of the 3-class classification model across different class-sampling proportions.** Metrics are computed from the thresholded multiclass evaluation. The 3-class confusion matrix is collapsed over the two positive classes (*cit.* and *cf.*); wrong positive-type predictions count as class-level errors. FPR, FNR, and SMR denote global error rates normalized by the total number of pairs N . Results are averaged across folds. Best value per column in bold.

Model	Verbatim (<i>cit.</i>)			Allusion (<i>cf.</i>)			Avg. F1
	Prec.	Rec.	F1	Prec.	Rec.	F1	
LatinBERT	0.77 ± 0.04	0.83 ± 0.03	0.78 ± 0.03	0.08 ± 0.03	0.42 ± 0.08	0.11 ± 0.03	0.45 ± 0.03
mmBERT Base	0.75 ± 0.02	0.75 ± 0.02	0.74 ± 0.02	0.09 ± 0.03	0.45 ± 0.08	0.13 ± 0.04	0.43 ± 0.02
mmBERT Small	0.72 ± 0.07	0.77 ± 0.04	0.72 ± 0.06	0.09 ± 0.02	0.37 ± 0.05	0.13 ± 0.03	0.42 ± 0.04
XLM-R Large	0.75 ± 0.06	0.81 ± 0.04	0.76 ± 0.04	0.07 ± 0.01	0.53 ± 0.09	0.11 ± 0.01	0.43 ± 0.02
XLM-R Base	0.69 ± 0.10	0.75 ± 0.04	0.69 ± 0.08	0.07 ± 0.02	0.56 ± 0.08	0.11 ± 0.03	0.40 ± 0.05
mBERT Large	0.71 ± 0.03	0.78 ± 0.03	0.72 ± 0.02	0.08 ± 0.01	0.36 ± 0.08	0.12 ± 0.02	0.42 ± 0.01
mBERT Base	0.69 ± 0.07	0.75 ± 0.03	0.69 ± 0.05	0.08 ± 0.02	0.42 ± 0.02	0.12 ± 0.02	0.41 ± 0.04
BERT-Romanian	0.62 ± 0.03	0.71 ± 0.03	0.63 ± 0.02	0.07 ± 0.03	0.42 ± 0.06	0.11 ± 0.04	0.37 ± 0.02
PhilBERTa	0.62 ± 0.09	0.73 ± 0.04	0.64 ± 0.07	0.03 ± 0.01	0.45 ± 0.10	0.05 ± 0.02	0.34 ± 0.04
LaBERTa	0.25 ± 0.17	0.66 ± 0.05	0.32 ± 0.15	0.02 ± 0.01	0.24 ± 0.06	0.03 ± 0.01	0.17 ± 0.08
RoBERTa-Latin	0.01 ± 0.00	0.26 ± 0.03	0.02 ± 0.00	0.01 ± 0.00	0.12 ± 0.04	0.02 ± 0.00	0.02 ± 0.00
LogReg (lexical)	0.37 ± 0.06	0.83 ± 0.03	0.46 ± 0.06	0.02 ± 0.01	0.37 ± 0.07	0.04 ± 0.01	0.25 ± 0.03
GBDT (lexical)	0.36 ± 0.04	0.84 ± 0.02	0.45 ± 0.04	0.02 ± 0.01	0.22 ± 0.07	0.04 ± 0.01	0.24 ± 0.02

Table 11: **Performance comparison of different pre-trained base models used in the classification stage.** Scores are given for *Verbatim (cit.)* and *Allusion (cf.)*; the last column is the macro-averaged F1. Per-class decision thresholds are tuned on the training split of each fold. Values are mean ± std over 5 cross-validation folds. The best value per column is bold and underlined.

k	# Predictions		Retrieval Only				Retrieve+Rerank				FPR × 10 ⁴		SMR × 10 ⁴	
	Ret.	Rer.	TP	FP	FN	F1	TP	FP	FN	F1	Ret.	Rer.	Ret.	Rer.
5	1,412	321	164	1,248	130	0.19	127	193	167	0.42	39.42	6.07	43.52	11.28
10	2,824	503	189	2,635	105	0.12	142	361	152	0.36	83.27	11.30	86.60	16.05
20	5,648	777	215	5,433	79	0.07	157	620	137	0.30	171.66	19.37	174.15	23.66
50	14,120	1,355	250	13,870	44	0.03	176	1,179	118	0.22	438.26	36.76	439.65	40.43
100	28,240	1,901	269	27,971	25	0.02	185	1,715	109	0.17	883.81	53.41	884.59	56.78
500	141,200	3,253	294	140,906	0	0.00	197	3,056	97	0.12	4452.32	94.03	4452.34	97.00
1000	159,489	3,347	294	159,195	0	0.00	197	3,150	97	0.12	5030.20	96.80	5030.20	99.76

Table 12: **Comparison between retrieval-only and retrieve-and-rerank (best retrieve-and-rerank configuration: BM25 + XLM-R Large).** k denotes the number of most similar candidates retrieved by BM25 that are reranked using the classification model. **Ret.** = total top- k candidates retrieved ($= k \times n_{\text{queries}}$); **Rer.** = positive predictions output by the reranker. Results are pooled across verbatim and allusion reference types and averaged across 5 folds. The reranking stage substantially reduces error rates (FPR, SMR), scaled by $\times 10^4$ for readability.

grows, the retrieval-only baseline reaches all 294 references only at $k \geq 500$, at the price of more than 140,000 false positives, whereas the pipeline’s false positives grow slowly (3,056 at $k = 500$) and its recall saturates at 197 of 294 references (67%): the remaining 97 are retrieved but rejected by the classifier, i.e., they are the classifier’s own misses analyzed in Appendix J. Beyond $k \approx 100$ the additional candidates add almost no true positives for the pipeline (185 to 197) but nearly double the number of flagged pairs (1,901 to 3,347), which is why the main text reports $k = 100$ as the practical operating point. Figure 12 shows the same trade-off for the two alternative configurations: replacing BM25 by BGE-M3 as the first stage lowers recall at every depth, in line with Table 2, and replacing XLM-R Large by the LogReg reranker keeps recall but forfeits most of the precision gain.

J Error Analysis

Table 13 reports the full per-author-pair, per-class breakdown for XLM-R Large on the pairwise evaluation grid. Unlike the compact table in the main error-analysis section, this appendix table includes precision, recall, F1, and the `locisimiles` error-rate metrics (FPR, FNR, SMR) for each author pair and each positive class.

The table should be read as a descriptive diagnostic, not as a pooled confusion matrix over the entire dataset. Each rate metric is computed within each fold using the `locisimiles` per-query macro definition in a one-vs-rest setup at $p \geq 0.5$, and the reported values are then averaged over folds. Support (#), by contrast, is summed across folds. Rows with very small support can therefore show extreme scores; the goal is to expose variation by author pair and reference type rather than to provide a second aggregate leaderboard.

The author-pair breakdown sharpens the pattern from the main text: verbatim performance tracks lexical overlap closely, while allusion performance is weaker and varies more by author pair. Jerome’s prose allusions remain especially difficult. The Valerius allusion rows, despite much lower overlap, benefit from larger support and a more homogeneous poetic source domain.

J.1 Qualitative Error Examples

To complement the quantitative breakdown, we inspect two representative cases in detail: a false positive labeled *cit.* and a false negative missed

entirely by the model.

False positive. Consider the false positive in Figure 13 (left). The lexical overlap is *omnia timeo* (“I fear everything”), and the verb *timere* (“to fear”) recurs in the source as *timenda sint* (“are to be feared”), strengthening the apparent signal. Yet *omnia timeo* is a very general formulation, carrying little distinctive content on its own. The failure is subtler: Jerome writes *ut ait gentilis poeta* (“as the pagan poet says”), signaling a genuine citation, yet it points to Virgil’s more specific *omnia tuta timens* (“fearing all things, even safe ones”). The crucial word *tuta* (“safe things”), absent from the matched source, is exactly what makes the Virgilian phrase distinctive. The model therefore identifies lexically significant overlap effectively, but it illustrates an edge case requiring expert analysis: the overlap points not to the proposed source but to a different one.

False negative. Inspecting a case missed entirely by the model (Figure 13, right) shows that divergent sentence context can be a problem. The source sentence is wholly concerned with the Socratic saying *nosce te* (“know yourself”). In the query, this reference appears only in the opening imperative *te ipsum intellege!* (“understand yourself!”); the larger part of the query then shifts to a discussion of stylistic simplicity, contrasting *verbosa rusticitas* (“wordy rusticity”) with *sancta simplicitas* (“holy simplicity”). The intertextual signal is therefore locally present but embedded in a broader context that points in a different thematic direction, which may explain why the model fails to retrieve the relevant source.

K Python Package

We release the framework described in this paper as an open-source Python package. The `locisimiles` library¹² implements the retrieve-and-rerank pipeline shown in Figure 4 and computes the task-specific error metrics (SMR, FPR, FNR) defined in Appendix B. It can be installed from PyPI¹³ with `pip install locisimiles`, and the API reference is available online.¹⁴

Because the package compares all segments of the query and source documents, evaluation must

¹²<https://github.com/julianschelb/locisimiles>

¹³<https://pypi.org/project/locisimiles/>

¹⁴<https://julianschelb.github.io/locisimiles/>

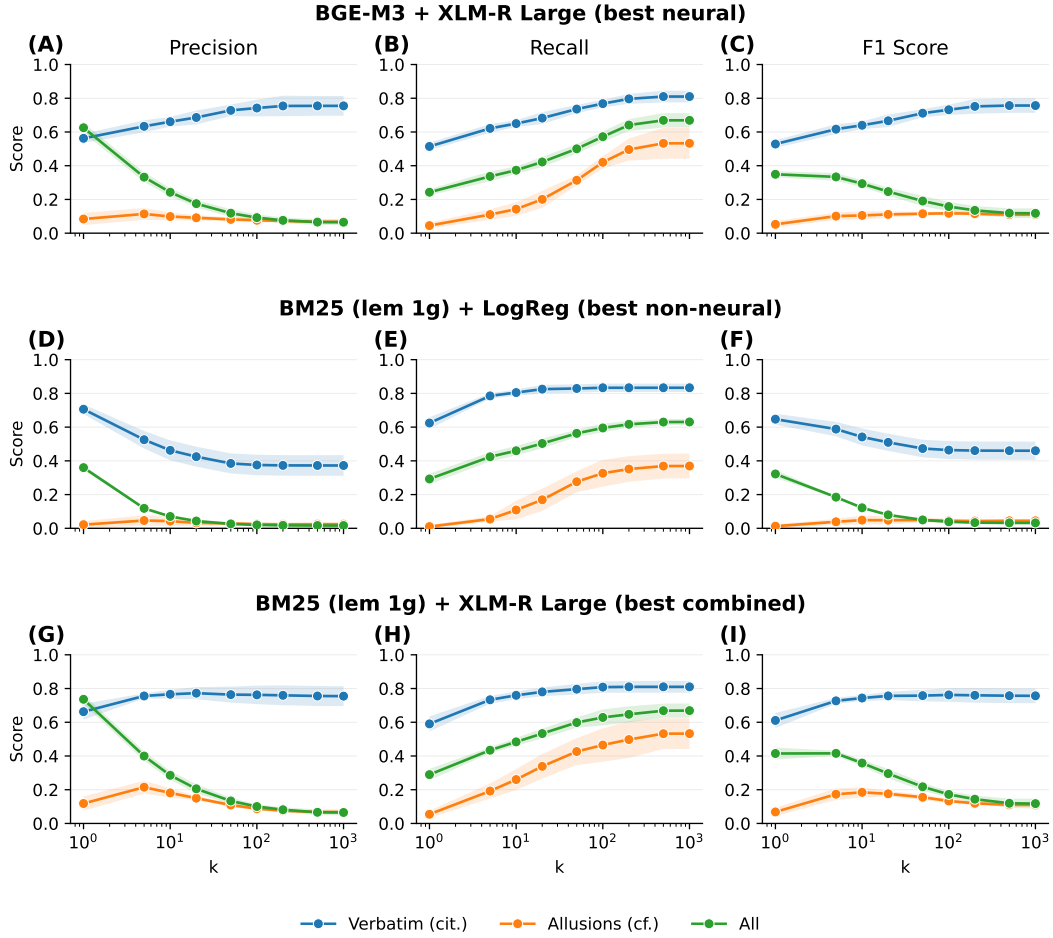


Figure 12: **Precision, Recall, and F1 Score of three retrieve-and-rerank pipeline configurations** as a function of the number of retrieved candidates k . Each row corresponds to one configuration: (*top*) BGE-M3 retrieval + XLM-R Large reranker, (*middle*) BM25 retrieval + LogReg reranker, (*bottom*) BM25 retrieval + XLM-R Large reranker. Within each panel, results are reported separately for verbatim references (*cit.*), allusions (*cf.*), and all references pooled (*all*). Shaded bands indicate ± 1 standard deviation across 5 cross-validation folds.

Query	Source	Verbatim (<i>cit.</i>)								Allusion (<i>cf.</i>)							
		#	Ov.	P	R	F1	FPR	FNR	SMR	#	Ov.	P	R	F1	FPR	FNR	SMR
Jerome	Cicero	91	5.22	0.651	0.673	0.656	0.003	0.000	0.004	21	3.14	0.050	0.033	0.040	0.001	0.000	0.001
Jerome	Horace	41	6.80	0.980	0.980	0.980	0.003	0.000	0.003	0	—	—	—	—	—	—	—
Jerome	Lucan	2	5.00	1.000	1.000	1.000	0.003	0.000	0.003	0	—	—	—	—	—	—	—
Jerome	Ovid	2	6.00	1.000	1.000	1.000	0.003	0.000	0.003	0	—	—	—	—	—	—	—
Jerome	Virgil	211	6.95	0.878	0.944	0.894	0.004	0.000	0.004	38	3.20	0.077	0.474	0.117	0.017	0.000	0.018
Lactantius	Horace	5	12.40	1.000	1.000	1.000	0.000	0.000	0.000	0	—	—	—	—	—	—	—
Lactantius	Lucretius	24	13.21	0.988	1.000	0.992	0.001	0.000	0.001	0	—	—	—	—	—	—	—
Lactantius	Ovid	23	11.13	0.980	1.000	0.987	0.001	0.000	0.001	0	—	—	—	—	—	—	—
Lactantius	Propertius	3	8.00	1.000	1.000	1.000	0.000	0.000	0.000	0	—	—	—	—	—	—	—
Lactantius	Virgil	78	8.85	0.894	0.983	0.917	0.001	0.000	0.001	0	—	—	—	—	—	—	—
Valerius	Lucan	65	1.91	0.529	0.522	0.518	0.002	0.003	0.005	85	1.33	0.236	0.641	0.324	0.078	0.003	0.081
Valerius	Ovid	55	1.86	0.540	0.511	0.521	0.001	0.001	0.002	94	1.16	0.214	0.609	0.294	0.047	0.002	0.049
Valerius	Statius	42	2.26	0.490	0.500	0.492	0.004	0.003	0.007	78	1.70	0.163	0.570	0.234	0.112	0.004	0.116
Valerius	Virgil	207	2.13	0.552	0.545	0.527	0.002	0.002	0.004	305	1.57	0.089	0.602	0.144	0.086	0.003	0.089
Total		849	5.11	0.755	0.810	0.757	0.003	0.000	0.003	621	1.65	0.067	0.533	0.110	0.017	0.000	0.017

Table 13: **Per-pair, per-class breakdown for XLM-R Large on the full pairwise evaluation set (mean across 5 folds).** Every rate metric (P, R, F1, FPR, FNR, SMR) is computed per fold using the `locisimiles` per-query macro definition (one-vs-rest at $p \geq 0.5$); support (#) is the sum across folds. Performance metrics (P, R, F1) are shaded teal (darker = larger), error rates (FPR, FNR, SMR) and overlap are shaded so that darker indicates larger values.

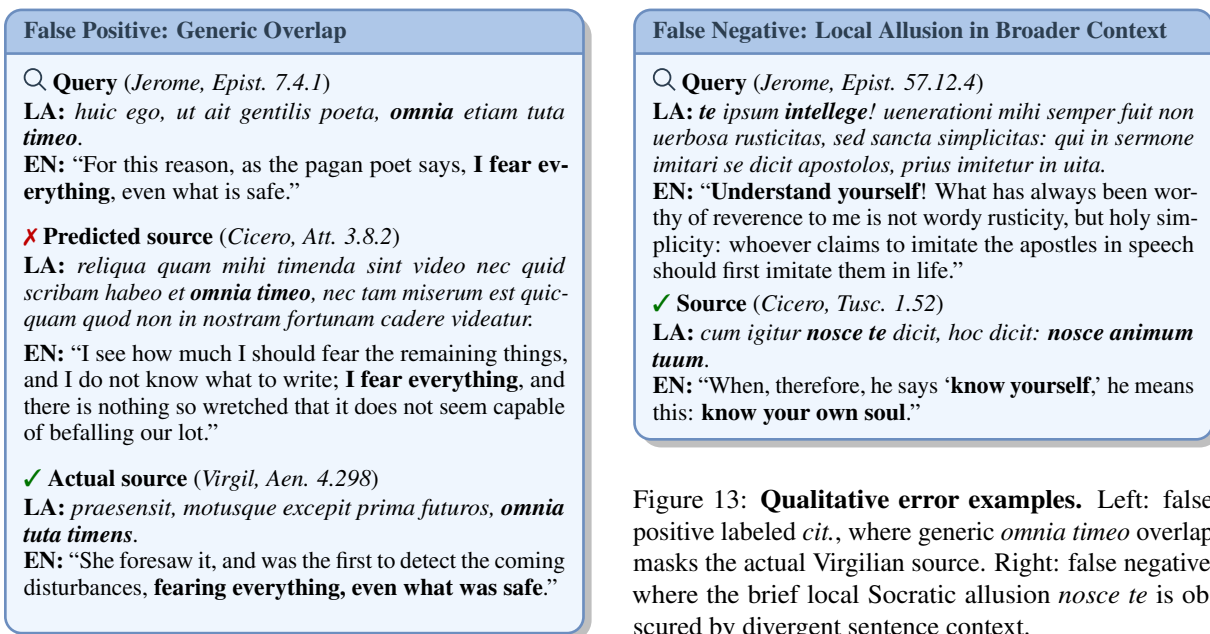


Figure 13: **Qualitative error examples.** Left: false positive labeled *cit.*, where generic *omnia timeo* overlap masks the actual Virgilian source. Right: false negative, where the brief local Socratic allusion *nosce te* is obscured by divergent sentence context.

Examples of Intertextual References		
Paraphrase (Minor)	Paraphrase (Major)	Allusion
SOURCE Cicero, Catil. 1.1 “Quo usque tandem abutere, Catilina, patientia nostra? ” (How long, Catiline, will you abuse our patience?)	SOURCE Virgil, Georg. 4.82 “... ingentes animos angusto in pectore versant ...” (... they wield mighty souls in a tiny breast ...)	SOURCE Cicero, Orat. 33.11 “... sed nihil difficile amanti puto.” (... but I think nothing is difficult for a lover.)
REUSE Jerome, Epist. 98.22.4 “... et patientia nostra quasi quodam temeritatis fomite abutentes ...” (... and abusing our patience like some kindling of rashness ...)	REUSE Jerome, Epist. 107.13.4 “... et in paruis corpusculis ingentes animos intueri!” (... and to see mighty souls in small bodies!)	REUSE Jerome, Epist. 22.40.1 “ Nihil amantibus durum est, nullus difficilis cupienti labor.” (Nothing is hard for lovers , no labor difficult for the desirous.)
Comment: Jerome integrates Cicero’s famous invective syntactically by adapting the verb form (<i>abutere</i> → <i>abutentes</i>).	Comment: Jerome retains the semantic core but rephrases <i>angusto in pectore</i> to <i>in parvis corpusculis</i> .	Comment: Jerome evokes the motif using synonymous but distinct vocabulary (<i>difficile amanti</i> vs. <i>amantibus durum</i>).

Figure 14: **Example references.** Three examples of text reuse by Jerome included in the ground-truth dataset.

account for the correct rejection of pairs labeled *no match* as well as the recovery of references. In addition to per-class precision (P), recall (R), and F1, *locisimiles* therefore reports three error rates normalized by the total number of segment pairs (N): SMR as a global error rate, FPR for spurious matches, and FNR for missed references. Standard accuracy is omitted because the extreme class imbalance makes it uninformative: a trivial all-negative predictor already achieves near-perfect scores. Formal definitions are given in Appendix B.

K.1 Python API

The core API lets researchers load custom query and source documents in CSV format and run the detection pipeline with our fine-tuned models from the Hugging Face Hub.¹⁵

```
# 1. Load query and source documents
query_doc = Document("query.csv")
source_doc = Document("source.csv")

# 2. Initialize the pipeline
pipeline = ClassificationPipeline(
    classification_name="...",
)

# 3. Run the pipeline
results = pipeline.run(
    query=query_doc,
    source=source_doc,
)

# 4. Display results
pretty_print(results)
```

K.2 Graphical User Interface

To lower the barrier to entry, the package includes an optional Gradio-based GUI. It can be installed via the optional dependency group (`pip install .[gui]`) and launched from the command line with `locisimiles-gui`.

Workflow. The application workflow is organized into three sequential stages, as illustrated in Figure 15:

1. **Data Upload:** Users upload custom query and source documents as CSV files.
2. **Configuration:** The pipeline is customized by selecting pre-trained models and tuning retrieval parameters (e.g., retrieval depth k and classification confidence thresholds).
3. **Result Exploration:** The interactive dashboard presents query segments alongside

retrieved source candidates, displaying cosine similarity and classification probability scores, with functionality to export confirmed matches.

The three panels show the upload view (Figure 15A), configuration view (Figure 15B), and result-exploration view (Figure 15C).

L Scalability and Future Directions

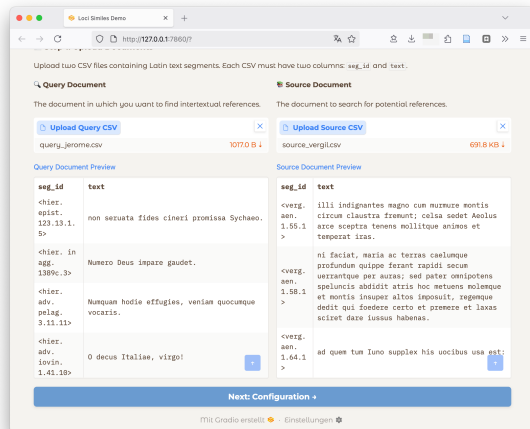
Scalability of the pipeline. The full corpus considered here contains roughly 7.7×10^9 possible query–source pairs. The retrieval stage reduces this space by orders of magnitude, so end-to-end cost is dominated by (i) the dense index built once per source corpus and (ii) cross-encoder passes on the top- k candidates per query. At larger corpus scales, neither component remains practical without dedicated indexing: dense retrieval must move from exact to approximate nearest-neighbor search, and the cross-encoder budget must be controlled by reducing k or by precomputing partial token representations. Mahadevan et al. (2025) discuss this billion-scale regime for historical text reuse and are the closest point of comparison for scaling *Loci Similes*.

Completeness of the ground truth. As noted in the Limitations, the expert-verified ground truth is not exhaustive. Some candidate pairs counted here as false positives may correspond to genuine but undocumented intertextual references, so precision figures on the curated subset should be read as lower bounds rather than as estimates over the complete reference graph.

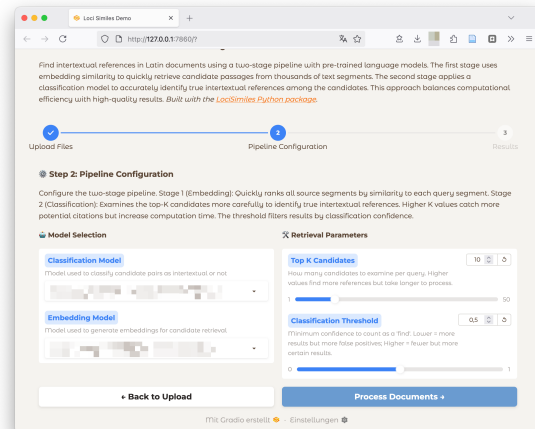
Directions opened by the benchmark. Beyond the modeling directions in Section 7, the benchmark supports two further lines of work: generative explanation, where a model produces a natural-language rationale for a proposed reference that can be checked against expert annotations for faithfulness; and indexing-aware retrieval, where late-interaction retrievers such as ColBERT (Khattab and Zaharia, 2020) and the data structures surveyed by Mahadevan et al. (2025) are evaluated under the same staged protocol, reporting quality and cost jointly.

¹⁵<https://hf.co/collections/julian-schelb/models-for-latin-intertextuality-search>

(A) Data Upload



(B) Configuration



(C) Interactive Result Exploration

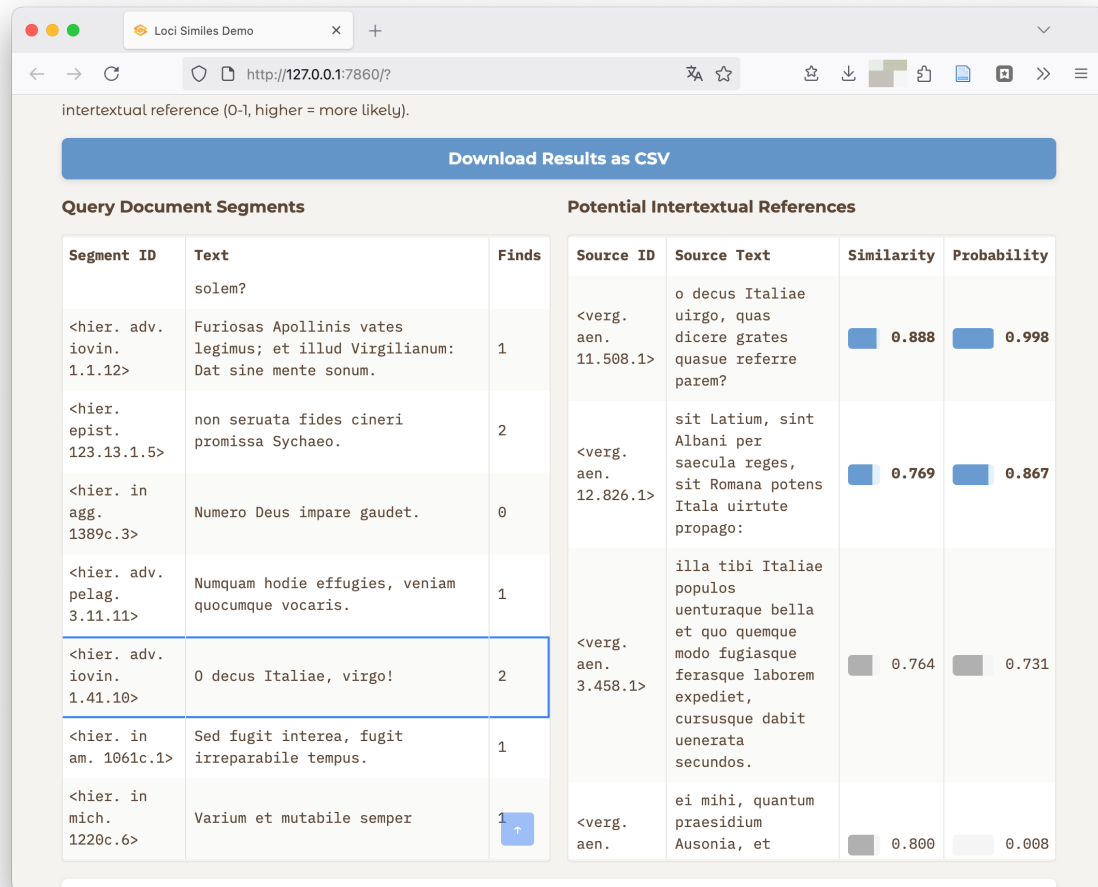


Figure 15: **Graphical user interface workflow.** (A) *Data Upload*: Users ingest query and source documents via CSV files. (B) *Configuration*: The pipeline is customized by selecting pre-trained models and tuning the retrieval depth (top- k) and classification confidence threshold. (C) *Result Exploration*: The dashboard presents query segments alongside retrieved source candidates, displaying both cosine similarity and classification probability scores, with options to export matches as a CSV file.