THIVLVC: Retrieval Augmented Dependency Parsing for Latin

Luc Pommeret[1], **Thibault Wagret**[2], **Jules Deret**
[1]Université Paris-Saclay, CNRS, LISN, 91400, Orsay, France
[2]École Normale Supérieure de Lyon, HISOMA, 69007, Lyon, France
pommeret@lisn.fr, thibault.wagret@ens-lyon.fr, deret.jules@gmail.com

Abstract

We describe THIVLVC, a two-stage system for the EvaLatin 2026 Dependency Parsing task. Given a Latin sentence, we retrieve structurally similar entries from the CIRCSE treebank using sentence length and POS n-gram similarity, then prompt a large language model to refine the baseline parse from UDPipe using the retrieved examples and UD annotation guidelines. We submit two configurations: one without retrieval and one with retrieval (RAG). On poetry (Seneca), THIVLVC improves CLAS by +17 points over the UDPipe baseline; on prose (Thomas Aquinas), the gain is +1.5 CLAS. A double-blind error analysis of 300 divergences between our system and the gold standard reveals that, among unanimous annotator decisions, 53.3% favour THIVLVC, showing annotation inconsistencies both within and across treebanks.

Keywords: Latin Dependency Parsing, Retrieval-Augmented Generation, Universal Dependencies, EvaLatin, Annotation Consistency

1. Introduction

The EvaLatin 2026 Dependency Task (Iurescia et al., 2026) invites participants to parse Latin texts for two genres: Classical poetry with Seneca, and philosophical prose of Thomas Aquinas.

Previous systems have relied on supervised neural models trained on existing treebanks. Such models learn whatever patterns the training data contains, including, inevitably, annotation choices that predate current guidelines. We explore a complementary approach: explicit injection of UD rules into a Large Language Model, allowing it to refine the output of a traditional parser, combined with a retrieval component. The method is simple. Whether it generalizes beyond Latin remains to be seen.

2. Description of the System

Our system is a two-stage pipeline[1]: (1) retrieval of structurally similar sentences from CIRCSE, and (2) generation, where an LLM refines a baseline parse using the retrieved examples and UD guidelines.

Information Retrieval. Given an input sentence, we retrieve the $k = 5$ most similar sentences from the training set of the CIRCSE treebank (762 sentences) using the *structural* retriever. This retriever is based on sentence length and POS n-grams provided in the input data. The similarity function is a weighted combination of normalized length difference and Jaccard similarities over POS bigrams and trigrams (Equation 1).

[1]Code available at https://github.com/l-pommeret/THIVLVC

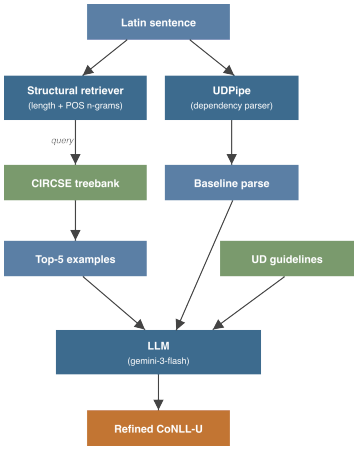

Figure 1: Overview of the THIVLVC pipeline. The input sentence is processed in parallel by the structural retriever (which selects similar examples from CIRCSE) and by UDPipe (which produces a baseline dependency parse). Both outputs, together with the UD guidelines, are passed to the LLM for refinement.

Generation. The retrieved examples, together with the official UD annotation guidelines and the baseline parse from UDPipe, are passed to `gemini-3-flash` (Google DeepMind, 2026). The LLM is prompted to act as a "Latin Chief Annotator": it compares the baseline parse with the retrieved examples and the guidelines, then outputs a refined CoNLL-U block with corrected HEAD and DEPREL columns. The full prompt is given in Appendix A.

We submit two configurations: THIVLVC_1 (LLM + UDPipe + UD guidelines, without retrieval) and THIVLVC_2 (same, but with RAG on CIRCSE). Figure 1 gives an overview of the pipeline.

Proceedings of LT4HALA 2026 @ LREC 2026, pages 219–225.
11 May 2026. ©ELRA Language Resources Association (ELRA), 2026

3. Evaluation Protocol

Retrieval strategies. We compared three retrieval strategies. In each case, q denotes the query sentence and s a candidate from the knowledge base.

1. TF-IDF (Baseline). Cosine similarity where $\mathbf{v}_q$ and $\mathbf{v}_s$ are the TF-IDF vectors of word forms for q and s:

$$\text{sim}_{\text{tfidf}}(q, s) = \frac{\mathbf{v}_q \cdot \mathbf{v}_s}{\|\mathbf{v}_q\| \, \|\mathbf{v}_s\|}$$

2. Structural (Length + POS n-grams). A weighted combination of sentence length similarity and POS n-gram Jaccard overlap:

$$\text{sim}_{\text{struct}}(q, s) = 0.33\, f_{\text{len}} + 0.33\, f_{\text{bi}} + 0.34\, f_{\text{tri}} \quad (1)$$

   where $f_{\text{len}} = 1 - \frac{||q|-|s||}{\max(|q|,|s|)}$ is the normalized length similarity ($|q|$ and $|s|$ denote sentence lengths in tokens), $f_{\text{bi}} = J(\text{bigrams}(\text{POS}_q), \text{bigrams}(\text{POS}_s))$ is the Jaccard coefficient $J(A, B) = |A \cap B|/|A \cup B|$ over POS bigrams, and f_{tri} is defined analogously for trigrams.
3. Morphological. Cosine similarity over TF-IDF vectors of concatenated POS and morphological features (`POS|FEATS` per token).

Retrieval metrics. Let $Q = \{q_1, \ldots, q_M\}$ be the test set. For each query q_i, we retrieve $k = 5$ examples $s_{i,1}, \ldots, s_{i,k}$ from the knowledge base. We evaluate retrieval quality with two metrics:

Length Difference: the average absolute difference in sentence length (in tokens):

$$\text{LenDiff} = \frac{1}{Mk} \sum_{i=1}^{M} \sum_{j=1}^{k} \big||q_i| - |s_{i,j}|\big|$$

A small difference (< 2 tokens) indicates similar syntactic complexity.

POS Overlap: the average Jaccard coefficient J (as defined above) over POS tag sets. For a sentence s, let $P(s) = \text{unique}(\text{POS}(s))$:

$$\text{POSOverlap} = \frac{1}{Mk} \sum_{i=1}^{M} \sum_{j=1}^{k} J\big(P(q_i),\, P(s_{i,j})\big)$$

Higher is better ($[0, 1]$).

Generation. We tested three recent LLMs of varying scale and provider: `gemini-3-flash`, `claude-4.5-sonnet`, and `qwen3-72B`.

Benchmarks. We use the CIRCSE treebank for retrieval evaluation and the EvaLatin 2026 test set for generation evaluation. The official metrics are CLAS and LAS (F1), reported both with and without relation subtypes.

Dataset	Strategy	Length Diff	POS Overlap
Prose	TF-IDF	13.24 ± 12.09	0.421 ± 0.150
	Morphological	12.86 ± 14.78	$\mathbf{0.519} \pm 0.138$
	Structural	$\mathbf{0.76} \pm 1.07$	0.512 ± 0.142
Poetry	TF-IDF	11.60 ± 15.95	0.454 ± 0.177
	Morphological	14.51 ± 20.63	0.556 ± 0.163
	Structural	$\mathbf{1.13} \pm 3.02$	$\mathbf{0.601} \pm 0.196$
Combined	TF-IDF	12.26 ± 14.52	0.441 ± 0.167
	Morphological	13.88 ± 18.56	0.541 ± 0.154
	Structural	$\mathbf{0.98} \pm 2.44$	$\mathbf{0.565} \pm 0.177$
CIRCSE	TF-IDF	11.63 ± 14.34	0.471 ± 0.185
	Morphological	15.55 ± 20.49	0.568 ± 0.175
	Structural	$\mathbf{1.06} \pm 1.73$	$\mathbf{0.611} \pm 0.198$

Table 1: Retrieval strategies comparison. Knowledge base: CIRCSE train (762 sentences). $k = 5$ retrievals per query.

4. Results and Analysis

IR results. Table 1 shows that the structural strategy strongly outperforms TF-IDF and morphological retrieval on length difference (< 1.2 tokens on average vs. > 11), while maintaining competitive POS overlap. This confirms that sentence length and POS n-grams are sufficient features for retrieving structurally similar examples.

System results. Table 2 compares our two configurations with the UDPipe baseline (Straka and Straková, 2020). On poetry, `THIVLVC_2` improves CLAS by +17 points over UDPipe (with subtypes). On prose, the gain is more modest (+1.5 CLAS), as UDPipe already performs well on this genre. The RAG component (`THIVLVC_2` vs. `THIVLVC_1`) brings a consistent improvement, especially on prose (+6.9 CLAS with subtypes).

		With subtypes		No subtypes	
System	**Genre**	**CLAS**	**LAS**	**CLAS**	**LAS**
`UDPipe`	Poetry	56.94	57.22	57.24	59.74
	Prose	79.41	82.17	83.07	85.21
`THIVLVC_1`	Poetry	72.71	70.36	76.00	76.97
	Prose	74.04	75.72	81.52	82.78
`THIVLVC_2`	Poetry	**74.03**	**72.88**	**76.08**	**77.60**
	Prose	**80.92**	**83.26**	**86.60**	**87.93**

Table 2: `THIVLVC` system comparison. `THIVLVC_1` = LLM + UDPipe + UD Guidelines. `THIVLVC_2` = `THIVLVC_1` + RAG on CIRCSE.

5. Error Analysis

Not all divergences from the gold standard are errors. To better understand our system's behaviour, we conducted a qualitative analysis of cases where predictions differed from reference annotations.

Annotation protocol. We designed a double-blind annotation comparing Gold and `THIVLVC`

outputs (see the interface in Figure 2). Annotators were presented with divergent annotations without knowing which came from which system, and chose among five categories: “Gold is better”, “System is better”, “both wrong”, “undecidable”, and “don’t know”.

Verdict	Ann. 1		Ann. 2	
	n	%	n	%
Gold (human)	137	45.7	110	36.7
THIVLVC (system)	126	42.0	143	47.7
Both wrong	7	2.3	11	3.7
Undecidable	11	3.7	21	7.0
Don’t know	19	6.3	15	5.0

Table 3: Blind evaluation verdicts per annotator (300 items).

Table 3 shows that out of 300 divergences, annotators unanimously agreed on 167 cases, of which 89 (53.3%) favour THIVLVC and 78 (46.7%) the gold standard (Table 4). Tables 5 and 6 in the Appendix break these down by error type and most frequent label confusions. Drawing on UD guidelines and recent work on Latin treebank harmonization (Gamba and Zeman, 2023), we organise the discussion into five categories.

6. Taxonomy of Disagreement

Contradictions between CIRCSE and EvaLatin. According to Table 6, errors of the type `advmod:lmod` instead of `advmod` account for 7 out of 37 (18%) of the main THIVLVC errors. The adverb *unde* illustrates a case of legitimate annotation divergence between the CIRCSE corpus and the EvaLatin 2026 corpus. In CIRCSE,[2] *unde* is annotated `advmod:lmod`, signalling a spatial reference in the underlying description. In EvaLatin 2026, by contrast, *unde* is annotated as a simple `advmod`, without the subtype `:lmod`.[3] Both analyses are linguistically defensible: *unde* may simultaneously carry a spatial origin reading (justifying `:lmod`) and function as a logical connector, and neither interpretation excludes the other. The two corpora simply reflect different (but individually coherent) annotation conventions regarding the scope of the `:lmod` subtype.

Our system reproduces the `advmod:lmod` pattern learned from the CIRCSE training data. When evaluated against EvaLatin 2026, this behaviour is penalized, although it reflects a valid annotation choice. This type of divergence suggests that evaluation metrics should ideally accept both annotations as correct when two conventions are independently defensible, rather than treating the test set annotation as the sole ground truth.

This case illustrates a broader methodological issue in dependency parsing: an annotation shift between training and evaluation datasets. Similar inconsistencies across treebanks have been documented in previous work on UD harmonization, especially among subtypes.[4] When a training corpus uses a subtype that the evaluation corpus omits, parsers may be penalized for faithfully reproducing the annotation patterns present in their training data.

Internal gold inconsistency: `obl` vs `obl:arg`. According to Table 6, errors of the type `obl` instead of `obl:arg` and `obl` instead of `obl:lmod` account for 12 out of 26 (46.2%) of the main EvaLatin 2026 errors. Some discrepancies between THIVLVC and EvaLatin 2026 arise from internal inconsistencies within EvaLatin 2026 itself, particularly in the use of the `obl:arg` subtype (the same goes for `obl:lmod`). EvaLatin 2026 contains divergent annotations for the verb *pertineo* and its prepositional complement introduced by *ad*.[5] The two examples receive different annotations despite their identical syntactic configuration. UD guidelines recommend `obl:arg` for oblique arguments selected by a predicate.[6]

THIVLVC, by contrast, annotates `obl:arg` in both cases, consistently with its training on CIRCSE and with the UD guidelines. The discrepancy therefore appears to result from variation in annotation practice rather than from a parsing error.

Such intra-corpus inconsistencies are distinct from the inter-corpus divergences discussed above. They suggest that EvaLatin 2026 contains a degree of internal annotation noise. Similar annotation fluctuations within the same dataset have been documented in previous work on UD harmonization (Gamba and Zeman, 2023). The distinction between arguments and oblique modifiers is widely recognized as difficult to apply consistently in dependency annotation (de Marneffe et al., 2014).

[2] Sentence `Latin_Tacitus_Ger_prose-163`: *unde annum quoque ipsum non in totidem digerunt species hiems et uer* [...] (“Hence they do not divide the year itself into the same number of seasons: winter, spring [...]”).

[3] E.g. sentence s142: *unde sacra doctrina maxime dicitur sapientia* (“Hence sacred doctrine is called wisdom in the highest sense”).

[4] “The most widespread issues are the `tmod` and `lmod` relation subtypes, as well as comparative clauses.” (Gamba and Zeman, 2023).

[5] In sentence S2, *propositum nostrae intentionis in hoc opere est, ea quae ad christianam religionem pertinent* [...], *religionem* is annotated `obl:arg` of *pertinent*. However, in sentence s147, *cum iudicium ad sapientem pertineat* [...], *sapientem* governed by the same verb with the same preposition, is annotated as bare `obl`.

[6] https://universaldependencies.org/u/dep/obl-arg.html

Clear-cut gold error: `mark` for `case`. Sentence s127 of EvaLatin 2026 illustrates a likely human annotation error.[7] The relation `mark` is reserved in the UD guidelines for subordinating conjunctions and clause-introducing function words (de Marneffe et al., 2014), whereas prepositions governing noun phrases are annotated as `case`. `THIVLVC` correctly annotates `case` in both instances, demonstrating consistency with the UD guidelines.

Similar misattributions of functional relations have been observed in Latin treebanks during harmonization and conversion processes. For instance, (Gamba and Zeman, 2023) report that nominal obliques were sometimes misannotated as `advmod` in PROIEL, while (Cecchini et al., 2018) show that the conversion of the *Index Thomisticus* from Prague Dependency style to UD introduced systematic errors in prepositional functional relations.

Undecidable ambiguity: *hic* (`det` or `advmod:lmod`?). Not all divergences between the parser and EvaLatin 2026 can be attributed to annotation error. Some reflect genuine linguistic ambiguity that the UD guidelines do not fully resolve.[8] Such cases illustrate a structural limitation of single-analysis treebank annotation. Because a gold standard records only one syntactic analysis, alternative interpretations are necessarily excluded, and parsers are penalized when they select a different but linguistically plausible structure. Cases of this kind have long been identified as a general challenge for treebank evaluation (Gamba and Zeman, 2023).

Errors linked with punctuation. Some groups of words presented as sentences in the corpus appear to contain several sentences.[9] If some editions read two sentences, the dependency between the root (*omitte*) and the head of the second clause (*traho*) could be analyzed as `parataxis` rather than `conj`. However, editorial punctuation reflects interpretive choices rather than established facts about the text, and the corpus annotation is not necessarily wrong for diverging from a given printed edition. EvaLatin 2026, however, reads `conj(omitte, traho)`, while `THIVLVC` has `parataxis(omitte, traho)`.

In another sentence, both EvaLatin 2026 and `THIVLVC` appear to make the same interpretation.[10] The lack of punctuation in the corpus can mislead both human annotators and automatic systems. A solution to this problem could be to better segment sentences in the corpus.

Adjectival misclassification: `amod` for `acl`. According to Table 6, errors of the type `amod` instead of `acl` account for 8 out of the 37 most frequent gold-correct cases (21.6%), making it one of the most frequent `THIVLVC` error types. In these instances, the system annotates past participles such as *subiecti*, *scissa* or *lapsi* as `amod` rather than `acl`, which is the annotation expected by the UD guidelines for participial constructions retaining verbal properties. The confusion is not entirely surprising from a linguistic standpoint: in traditional grammar, the participle is conventionally described as the adjectival form of the verb. The case of *notus* illustrates this ambiguity well: originally a participle of *nosco*, it has undergone a degree of lexicalisation and functions in many contexts as a plain adjective ("well-known"), making either analysis defensible. Nevertheless, the systematic nature of the misinterpretation (affecting multiple participles across different sentences) remains difficult to account for. The CIRCSE training data consistently distinguishes `acl` from `amod` in comparable configurations, and the UD guidelines passed to `THIVLVC` explicitly define `acl` as the appropriate relation for participial modifiers of nominals.

7. Limitations

Our approach has several limitations. First, the selection of the LLM was based on informal manual comparison rather than a systematic ablation study. We tested `gemini-3-flash`, `claude-4.5-sonnet`, and `qwen3-72B` on a small set of sentences and selected `gemini-3-flash` on the basis of output quality and cost, but we did not conduct a controlled evaluation across models. This limits the reproducibility and generalizability of our LLM choice. Second, the system relies on a single

[7] In *sed haec doctrina supponit principia sua aliunde, ut ex dictis patet*, the word *ex* is annotated as `mark`. The same preposition is correctly annotated as `case` in sentence s47: *omnis enim scientia procedit ex principiis per se notis*.

[8] In the CIRCSE sentence *hic rapax torrens cadit* [...] (SenPhoen-P-17-8), the form *hic* can be analyzed either as an anaphoric determiner modifying *torrens* (`det`: "this violent torrent here falls") or as a locative adverb attached to *cadit* (`advmod:lmod`: "here a violent torrent falls"). Both readings are syntactically and philologically defensible.

[9] For example, *omitte poenae languidas longae moras mortemque totam admitte quid segnis traho quod uiuo* (SenPhoen-P-17-22, Loeb: "Away with the slow delays of thy long-due punishment; receive death wholly. Why do I sluggishly drag on this life?"). Both the Latin text and the translation distinguish two sentences, separated by a full stop, also in the Teubner edition.

[10] In *flammas potius et uastum aggerem compone in altos ipse me immittam rogos* [...] (SenPhoen-P-17-61), both annotate `conj`(*compone*, *immittam*). The Loeb edition, however, distinguishes the two clauses with a semicolon, suggesting `parataxis`(*compone*, *immittam*).

knowledge base (CIRCSE, 762 sentences). The retriever cannot return useful examples for syntactic constructions absent from this small corpus, which may explain the more modest gains on prose, where UDPipe already performs well. Third, LLM-based parsing is inherently non-deterministic: the same prompt can yield different outputs across runs. We did not measure this variance. A related concern is cost: each sentence requires an API call, making the approach substantially more expensive than a fine-tuned neural parser. Finally, our error analysis, while informative, is limited to 300 items annotated by two annotators with fair inter-annotator agreement ($\kappa = 0.49$ on the binary Gold/`THIVLVC` decision). A larger annotation campaign would be needed to draw firmer conclusions.

8. Conclusion

We presented `THIVLVC`, a retrieval-augmented LLM system for Latin dependency parsing that combines a structural retriever, UD annotation guidelines, and a baseline parse to refine syntactic analysis. The system achieves substantial improvements over UDPipe on poetry and competitive results on prose. Our error analysis highlights a finding that goes beyond system performance: a significant proportion of divergences between parser output and gold annotations stem from annotation inconsistencies, both across corpora (CIRCSE vs. EvaLatin 2026) and within a single dataset. These results underscore the need for continued harmonization efforts in Latin treebanks, in line with previous work (Gamba and Zeman, 2023). More broadly, our analysis suggests that evaluation protocols for dependency parsing could benefit from accepting multiple valid annotations in cases where legitimate annotation conventions diverge or where genuine linguistic ambiguity makes a single gold standard inadequate. In future work, we plan to conduct a systematic comparison of LLMs for this task, expand the retrieval knowledge base to multiple treebanks, and investigate whether fine-tuning a smaller model on LLM-generated corrections could reduce inference cost while preserving quality.

9. Acknowledgements

We thank the EvaLatin 2026 organizers for making the shared task data available, and the CIRCSE Research Centre for the treebank used as knowledge base. This work was carried out at LISN (CNRS, Université Paris-Saclay) and HISOMA (ENS of Lyon).

10. Bibliographical References

Flavio Massimiliano Cecchini, Marco Passarotti, Paolo Ruffolo, Marinella Testori, Lia Draetta, Martina Fieromonte, Annarita Liano, Costanza Marini, and Giovanni Piantanida. 2018. Enhancing the Latin morphological analyser LEMLAT with a medieval Latin glossary. In *Proceedings of the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018)*, pages 88–93, Turin, Italy. CEUR Workshop Proceedings.

Marie-Catherine de Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D. Manning. 2014. Universal Stanford dependencies: A cross-linguistic typology. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4585–4592, Reykjavik, Iceland. European Language Resources Association (ELRA).

Federica Gamba and Daniel Zeman. 2023. Universalising Latin Universal Dependencies: a harmonisation of Latin treebanks in UD. In *Proceedings of the Sixth Workshop on Universal Dependencies (UDW, GURT/SyntaxFest 2023)*, pages 7–16, Washington, D.C. Association for Computational Linguistics.

Google DeepMind. 2026. Gemini 3 flash: A large multimodal model. `https://deepmind.google/technologies/gemini/`. Consulté le 10 mars 2026.

Federica Iurescia, Marco Passarotti, and Rachele Sprugnoli. 2026. Overview of the Dependency Parsing Task at EvaLatin 2026. In *Proceedings of the Fourth Workshop on Language Technologies for Historical and Ancient Languages (LT4HALA 2026)*, Palma, Mallorca (Spain). ELRA.

Milan Straka and Jana Straková. 2020. UDPipe at EvaLatin 2020: Contextualized embeddings and treebank embeddings. In *Proceedings of LT4HALA 2020 - 1st Workshop on Language Technologies for Historical and Ancient Languages*, pages 124–129, Marseille, France. European Language Resources Association (ELRA).

A. Prompt

The LLM receives the following prompt (the variables are filled at runtime):

```
You are the Latin Chief Annotator.
Your goal: Refine and improve the syntax
(HEAD/DEPREL) of the Input Sentence using
best practices.

=== OFFICIAL ANNOTATION GUIDELINES ===
{guidelines_text}
======================================

Here are 5 SIMILAR examples from the
training data.
{example_str}

--- BASELINE (from automatic parser) ---
{latinpipe_context}

--- Input Sentence ---
{conllu_str}

Task:
1. Compare the baseline parse with the
   examples and guidelines
2. Identify any improvements needed in
   HEAD/DEPREL
3. Output your refined version
4. If uncertain, add a comment line
   # needs_council = true

Output ONLY the CoNLL-U block for the
Input Sentence (not the baseline).
```

B. Annotation Interface

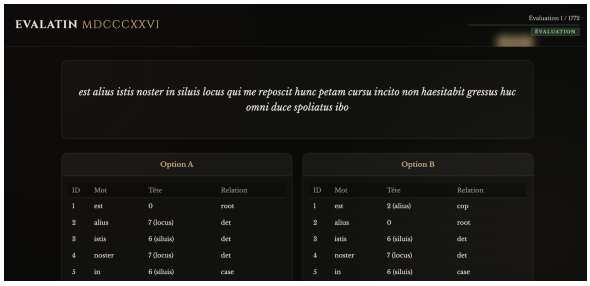

Figure 2: Annotation interface: overview. The Latin sentence is displayed at the top; two anonymized parse options (A and B) are shown side by side with their ID, word form, head, and relation columns.

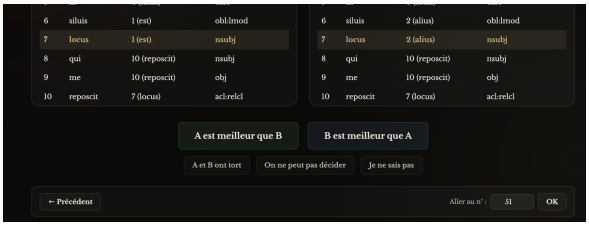

Figure 3: Annotation interface: verdict buttons. Divergent rows are highlighted. Annotators choose among five categories.

C. Detailed Annotation Results

We provide detailed statistics to support the comparison between THIVLVC and the gold standard.

Table 4 shows that, overall, THIVLVC is more often correct than the gold in head-to-head comparisons, at least on items where both annotators reach a consensus.

Table 4: Unanimous consensus: items where both annotators agree (173/300, 57.7%).

Verdict	n	%
THIVLVC better than Gold	89	53.3
Gold better than THIVLVC	78	46.7
Decided	*167*	
Undecidable	3	1.7
Don't know	3	1.7
Total	**173**	

When examining the different error types in Table 5, we observe that the most frequent errors are distributed across head attachment, relation labelling, and subtype confusion.

Table 5: Taxonomy of errors by source (Gold vs. THIVLVC), based on 167 unanimous decided cases.

Error type	**Gold err.**		**THIVLVC err.**	
	n	%	n	%
Wrong head + relation	26	29.2	14	17.9
Subtype confusion	23	25.8	29	37.2
Wrong head only	20	22.5	8	10.3
Wrong relation only	17	19.1	24	30.8
Head + subtype	3	3.4	3	3.8
Total	**89**		**78**	

The label confusions are also informative: Table 6 shows that the most frequent gold errors involve obl, while the most frequent THIVLVC errors involve amod and advcl.

Table 6: Most frequent label confusions by error source.

Gold errors (THIVLVC correct)			**THIVLVC errors (Gold correct)**		
Wrong	Correct	n	Wrong	Correct	n
obl	obl:arg	6	amod	acl	8
obl	obl:lmod	6	advcl	advcl:cmp	8
amod	root	4	advmod	advmod:tmod	7
obl	nummod	4	advmod:lmod	advmod	7
conj	parataxis	3	advmod	discourse	4
obl:agent	obl:arg	3	conj	conj:expl	3

Regarding genre, Table 7 shows that the consensus is fairly evenly distributed between poetry and prose.

Table 7: Consensus results by genre (167 decided cases).

Genre	**Gold**	**THIVLVC**	**Total**	**THIVLVC %**
Poetry	49	50	99	50.5
Prose	29	39	68	57.4
All	**78**	**89**	**167**	**53.3**

Table 8 shows that inter-annotator agreement is reasonably solid, with a Cohen's κ of 0.493 on head-to-head Gold vs. THIVLVC items.

Table 8: Inter-annotator agreement on 300 blind evaluation items. κ is Cohen's kappa.

Metric	**All categories**	**Gold/THIVLVC only**
Items evaluated	300	224
Observed agreement (p_o)	0.577	0.746
Expected agreement (p_e)	0.374	0.499
Cohen's κ	0.324	0.493

Table 9 presents the full confusion matrix between annotators, showing that most disagreements arise when one annotator selects Gold and the other selects THIVLVC.

Table 9: Confusion matrix between annotators (300 items). Rows = Ann. 1, Columns = Ann. 2.

	Gold	**THIVLVC**	**Both**	**Undec.**	**DK**	**Total**
Gold	**78**	37	6	8	8	137
THIVLVC	20	**89**	3	10	4	126
Both	3	4	0	0	0	7
Undec.	3	5	0	**3**	0	11
DK	6	8	2	0	**3**	19
Total	110	143	11	21	15	300