LLM as a Morphological Disambiguator for Belarusian: A Preliminary Study

Vladislav Poritski[1], Oksana Volchek[1], Ilia Afanasev[2]
[1] Independent researcher, Vilnius, Lithuania
[2] University of Vienna
v.poritski@gmail.com, volchekoa@gmail.com, ilia.afanasev.1997@gmail.com

Abstract

We explore the use of large language models (LLMs) for morphological disambiguation in Belarusian, a low-resource language. The pipeline has two stages: a rule-based analyzer generates candidate lemmas and grammatical tags, which an LLM then disambiguates in context. Initial evaluation of ChatGPT, Claude, and Gemini on a gold-standard sample shows high accuracy. We scale this approach to a 375K-word corpus using Gemini and compare the results against a neural baseline (Stanza). Manual review of discrepancies suggests that the LLM-based approach outperforms the baseline, offering a solution for corpus annotation in Belarusian.

Keywords: large language models, morphological disambiguation, Belarusian

1. Introduction

Emergent multilingual capabilities of large language models (LLMs) open the way for their use as co-annotators in developing language resources for under-resourced languages (Kholodna et al., 2024; Nahum et al., 2025). Improving annotation quality in existing silver-standard datasets (Wissler et al., 2014) is one potential use case for LLMs. However, in order to explore the limits of their applicability, careful task-specific and language-specific evaluation is necessary.

In this short paper, we consider the task of morphological disambiguation for Belarusian – a low-resource, morphologically rich East Slavic language, closely related to Russian and Ukrainian but spoken by a much smaller population and represented on the web less widely, by orders of magnitude. For a word token with sufficient context around it, the task is to choose the correct lemma and tag from the lists of candidates. This is relevant because, as of yet, the amount of publicly available morphologically disambiguated Belarusian data is rather small: e.g., the treebank UD_Belarusian-HSE (Shishkina and Lyashevskaya, 2021) in Universal Dependencies (Nivre et al., 2020) contains 305K tokens in 25K sentences. In larger corpora, such as the National Corpus of the Belarusian Language[1], the morphological annotation layer is derived from morphological databases like GrammarDB[2] without disambiguation.

To evaluate the feasibility of LLM-powered disambiguation, we use YABC[3], a freely downloadable corpus of ≈7.5M words of Belarusian newspaper articles and fiction. Developed in 2012–2014 and thus pre-dating the Universal Dependencies format, the corpus uses monolithic grammatical tags produced by a custom rule-based analyzer[4]. Even though its database is smaller than GrammarDB, the YABC morphological analyzer recognizes ≈98% tokens in the corpus, producing candidate lemmas and tags for them. We hypothesize that, with this information as input, a state-of-the-art LLM could be prompted to output high-quality annotation, matching or surpassing the quality delivered by statistical tools trained on UD_Belarusian-HSE.

We make three contributions. In § 3.1, we assess the performance of current LLMs in disambiguating Belarusian text, using a small manually disambiguated sample as reference. In § 3.2, we disambiguate a subset of YABC with a strong model and analyze the quality by reviewing samples of annotated data. In § 3.3, we compare how the two-stage approach (candidate generation + LLM disambiguation) fares against a one-stage neural lemmatization and PoS tagging baseline.

2. Related work

Starting from GPT-3, the research community has been exploring the potential use of LLMs as data annotators (Ding et al., 2023). Successful application to English data has been reported in text classification tasks (Gilardi et al., 2023), pragmadiscursive corpus annotation (Yu et al., 2024), and grammatical annotation (Morin and Marttinen Larsson, 2025). Beyond English, the results are mixed, even in high-level tasks like text classification (Bhat and Varma, 2023; Nasution and Onan, 2024).

[1] https://bnkorpus.info/index.en.html
[2] https://github.com/Belarus/GrammarDB
[3] https://github.com/poritski/YABC
[4] https://github.com/volchek/beltagger (latest version, a rewrite of the original tool).

Proceedings of SIGUL 2026 Joint Workshop with ELE, EURALI, and DCLRL @ LREC 2026, pages 42–48
11–12 May 2026.

Few attempts have been made so far to apply LLMs to morphological tagging tasks, in particular to lemma and tag disambiguation – possibly because the accuracy of statistical PoS taggers has long ago approached the level of human inter-annotator agreement (Manning, 2011), making the use of an even more capable general-purpose tool seem like overkill. In a systematic study that covers 247 morphosyntactic probing tasks in 42 languages from Universal Dependencies, Ács (2025, § 6.5) found that average zero-shot accuracy of GPT-4o is on par both with the neural tagger Stanza (Qi et al., 2020) and supervised mBERT and XLM-RoBERTa baselines, but the distribution of errors is different: for example, GPT-4o is better at determining nominal gender and worse at determining case. In a low-resource setting, Hämäläinen (2024) used GPT-4o to disambiguate lemmas in two endangered Uralic languages, Erzya and Skolt Sami, in which the model is not proficient. Augmenting the prompt with translations of the candidate lemmas into a related higher-resourced language, Finnish, resulted in 50% and 41% sentence-level accuracy respectively. Most recently, Vidal-Gorène et al. (2026) found both GPT-4o and an open-weight model Mistral Large to outperform a neural baseline in lemmatization and PoS tagging of the historical varieties of Greek, Armenian, Georgian, and Syriac.

Unlike Ács, who predicts one grammatical feature value at a time given one sentence as context, and Hämäläinen, who disambiguates all lemmas in one sentence at a time, we use larger batches, disambiguating all lemmas and tags in several paragraphs at once. We run the same batch multiple times through the model with non-deterministic decoding and then choose the best candidate lemmas and tags by majority vote; this approach falls into the paradigm of self-consistency (Wang et al., 2023). In tasks where providing wider context to the model facilitates more accurate judgments, it has been argued that batching can amplify the benefits of self-consistency (Korikov et al., 2025).

3. Experiments

3.1. LLMs vs. manual disambiguation

The authors of YABC shared one manually disambiguated document with us – an article around 1000 word tokens long, published by the daily newspaper *Źviazda* in 2008. Lemmas and grammatical tags were disambiguated by a paid annotator, a fourth-year B.A. student who had previously completed a similar unpaid pre-screening task.

We wrote a detailed prompt (see Appendix A) that includes an explication of the tagset and a disambiguation instruction with examples. The prompt instructs the model to:

- not modify any tokens and not change the tokenization, even if it appears incorrect;
- suggest appropriate lemmas and tags for tokens not recognized by the analyzer;
- choose the best candidate lemmas and tags among those returned by the analyzer, or suggest its own candidates but only if the model is absolutely confident that the correct lemma / tag is missing from the candidate list.

With this prompt we ran the non-disambiguated document through three proprietary LLMs available free of charge via their web UIs: ChatGPT (using the "think longer" feature), Claude 4.5 Sonnet and Gemini 3 Flash Preview. The input was formatted as tab-separated values, with one word token per line alongside its pipe-delimited candidate lemmas and tags. Each response was elicited n=3 times at default settings, with at least weekly intervals between runs to avoid response caching.

The accuracy of lemma and tag disambiguation by each model with respect to the reference is shown in Table 1. Claude and Gemini perform equally well, while ChatGPT is slightly weaker and notably less consistent, probably because it uses dynamic model routing under the hood. All models disambiguate tags less accurately than lemmas. Majority voting yields accuracy above the arithmetic average across the runs.

3.2. Disambiguating a larger sample

To disambiguate a larger subset of YABC, we chose Gemini 3 Flash Preview (Google DeepMind, 2025), due to its good performance and the availability of a free API quota – 20 requests per user per day at the time of our experiments (Jan–Feb 2026). We used this quota for the runs described below.

YABC includes a collection of 476 articles (≈375K word tokens) published in 2008–2009 by the youth newspaper *Čyrvonaja źmiena*, a biweekly supplement to the newspaper *Źviazda*. We corrected several hundred tokenization errors and processed the texts once again with the latest version of the YABC morphological analyzer. The analyzer did not recognize 1.37% of the word tokens. For each recognized token, on average there are 1.08 candidate lemmas and 1.80 candidate grammatical tags. The true rate of ambiguity is in fact higher because the correct candidate is sometimes missing from the analyzer's output (e.g., proper names misidentified as common nouns).

The dataset was split by paragraph boundaries into 188 batches, each around 2000 word tokens long (except the underfull last batch), and processed n=3 times with Gemini 3 Flash Preview via its API at default settings (temperature: 1.0, thinking level: high). We re-used the prompt from § 3.1. In addition to $188 \cdot n = 564$ successfully processed batches, there were 96 failures (truncated outputs,

Model	Lemma			Tag		
	runs 1–3	maj.	best	runs 1–3	maj.	best
ChatGPT	0.967 / 0.975 / 0.972	0.973	0.983	0.892 / 0.913 / 0.850	0.889	0.938
Claude 4.5 Sonnet	0.989 / 0.988 / 0.989	0.989	0.994	0.928 / 0.943 / 0.927	0.935	0.951
Gemini 3 Flash Preview	0.985 / 0.989 / 0.985	0.988	0.994	0.933 / 0.940 / 0.947	0.944	0.955

Table 1: Accuracy of lemma and tag disambiguation by LLMs in a sample of Belarusian texts from YABC, evaluated against a manually disambiguated reference. Majority-vote accuracy is the ratio of tokens for which the correct lemma / tag is the most frequent prediction across the runs. Best possible accuracy is the ratio of tokens for which the correct lemma / tag has been predicted in at least one run.

formatting errors that are not easily recoverable, etc.), yielding a success rate of 85.5%. The statistics of input and output sizes, measured in Gemini tokens, are shown in Table 2.

	Per batch	Per run
Input	33.0K. . .41.6K	7.01M
Output	21.4K. . .26.1K	4.37M (avg.)

Table 2: Gemini tokens spent on processing the *Čyrvonaja źmiena* dataset. Per-batch statistics do not include the last batch. Per-run statistics do not include failed batches.

Upon receiving the model's responses, we edited them semi-automatically (programmatic error detection with manual and programmatic error correction), maintaining the following invariants:

- The output lines are in one-to-one correspondence with the input lines.
- Each output line has three tab-separated fields (word token, lemma, tag) with no additional whitespace characters around the separators.
- The word token in the output line is the same as in the corresponding input line.
- Lemmas / tags outside the candidate lists were suggested for <5% of the tokens in the batch (or else the response was regenerated).

On average, 0.3% of all characters in a raw output batch are removed or added. The majority of edits are related to formatting. Tabs instead of newlines are the most common formatting error in Gemini outputs. In tokens copied from input to output with errors, the most common modifications are Latin transliteration (*на* 'on' → *na*) and translation into English (*з* 'with' → *with*), Russian or Ukrainian (*пры* 'at' → *при* 'at [in Russian / Ukrainian]'; *ад* 'from' → *від* 'from [in Ukrainian]').

After ensuring proper format, we reviewed some of the model's outputs, focusing on four classes described in Table 3. Except the first of them, where majority voting isn't possible, we picked a random 100-instance sample from each class (here an instance is a token with surrounding context) and manually labeled the model's majority-vote lemmas / tags as correct, wrong, or unclear, i.e. convention-dependent. Not covered in our analysis is the (arguably trivial) class of tokens with only one candidate lemma / tag, copied from input to output.

The classes of tokens with **no majority vote** and tokens **not recognized by the analyzer** intersect. 30% of the tokens without a majority-vote lemma and 25% without a majority-vote tag are unrecognized. Other no-majority-vote instances are highly ambiguous: 1.82 candidate lemmas,[5] 3.66 candidate tags per token on average. Conversely, among unrecognized tokens the ratios of no majority vote are relatively high: 0.88% lemmas (22x average), 3.76% tags (19x average), i.e., predicting is much less consistent than disambiguating.

For tokens without a majority-vote lemma, determining the lemma is often convention-dependent, such as in comparatives (*больш* 'more' – lemma *больш* or *многа* 'many') and participles, or it may be tricky for a human as well, such as in pronouns (possessive *іх* 'their' – lemma *іх*; personal *іх* 'they.GEN' – lemma *яны* 'they.NOM'). Pronominal words are also frequent among tokens without a majority-vote tag, along with proper and common nouns.

Unrecognized tokens that received majority-vote tags are most frequently analyzed as proper nouns (43.2%) and common nouns (35.0%). Unlike the YABC morphological analyzer, GrammarDB would recognize some of these but its coverage is uneven: e.g., among proper names toponyms are covered better (52.6%) than people's first names (29.3%) and surnames (8.5%).

Lemmas / tags **outside the candidate list** pose many problems for manual review, as the analysis is often convention-dependent (lemmas and tags in pronominal words, lemmas in comparatives and participles). Proper names, including first name initials, are the largest group of instances with correct majority-vote lemmas and tags in this class. Incorrect lemmas are systematically predicted for some adjectives and two verbs: *разумець* 'to understand' and *ставіцца* 'to treat'. Correct majority-vote tags are predicted for nonstandard case and num-

[5] With one or two candidate lemmas, instances of no majority vote occur because the model occasionally hallucinates, introduces orthographic errors, or defaults to the raw token.

Name	Explanation	Frequency	Sample +	−	?
No majority vote	Tokens for which the model predicts a different lemma / tag in each run	0.04% lemmas 0.20% tags	N/A		
Not recognized by the analyzer	Tokens for which the model doesn't have any candidates to choose and has to predict lemmas / tags on its own	1.37% lemmas 1.37% tags	91 81	1 4	8 15
Outside the candidate list	Tokens for which the model's majority-vote lemma / tag is not among the candidates proposed by the analyzer	0.29% lemmas 0.84% tags	51 76	18 13	31 11
Candidate choice	Tokens for which the model's majority-vote lemma / tag is one of ≥2 candidates proposed by the analyzer	7.42% lemmas 37.76% tags	73 90	11 5	16 5

Table 3: Nontrivial classes of LLM disambiguation outputs. Frequency of a class is the ratio of its token count to the total size of the dataset. 100-instance random samples are labeled manually (+: the majority-vote lemma / tag is correct, −: wrong, ?: unclear or convention-dependent).

ber forms of common nouns (*з пункту гледжання* 'from point.~~DAT~~ GEN.SG of view'). Assigning gender to the pronoun *што* 'what' is a common error in majority-vote tags.

The overall quality of **candidate choice** is good. Majority-vote lemmas are occasionally wrong for possessive (*яго* 'his'), demonstrative (*гэты* 'this'), total pronouns (*увесь* 'all'), suppletive nouns (*людзі* 'people' – correct lemma *чалавек* 'person'). Wrong tags highlight gaps in understanding case and gender agreement. For example:

ягоны *будынак* *збіраюцца* *знесці*
his.~~NOM~~ ACC.SG building plan.PRS.3PL demolish
'They plan to demolish his building.'

журы, у склад якога *ўваходзілі*...
jury.N in which.~~M~~ N.GEN.SG enter.PST.3PL
'Jury that consisted of...'

3.3. Comparison with one-stage baseline

Stanza (Qi et al., 2020), a popular multilingual NLP package, offers a neural pipeline of lemmatization and PoS tagging for Belarusian, pre-trained on UD_Belarusian-HSE data. We used it as a baseline against which to evaluate our two-stage approach to disambiguating the *Čyrvonaja źmiena* dataset.

We sentence-tokenized the texts. In non-Latin tokens, occurrences of Latin *i* (U+0069) were replaced with Cyrillic *і* (U+0456). Each sentence with its original word tokenization (`tokenize_pretokenized=True`) was then lemmatized and PoS tagged with Stanza. To ensure tagset compatibility, we created a mapping of all ⟨UPOS, XPOS, features⟩ triples that occur in Stanza outputs with frequency >1 (which amounts to ≈99.95% of the tokens) into YABC tags. The mapping is not always one-to-one: e.g., Universal Dependencies features do not encode verb transitivity, and YABC tags do not encode noun animacy. We consider the Stanza tag to differ from its respective majority-vote tag if the set of valid YABC tags derived from the former does not include the latter.

Statistics of discrepancies between the majority-vote lemmas / tags and their baseline counterparts are provided in Table 4.

	Frequency	Sample +	−	?
Lemma	6.29%	55	15	30
Tag	10.53%	68	15	17

Table 4: Discrepancies between the LLM majority-vote outputs and the neural baseline. 100-instance random samples are labeled manually (+: LLM is better, −: Stanza is better, ?: unclear).

Among Stanza lemmas, verb infinitives are sometimes spelled in *taraškievica* (an older literary standard of Belarusian), likely because >40% of the sentences in UD_Belarusian-HSE use this spelling. Stanza correctly lemmatizes some of the instances known to be hard for Gemini, such as the inflected forms of *людзі* and *разумець*. Correct grammatical tags are sometimes predicted for possessive pronouns and predicative words, mis-analyzed by Gemini. However, the disambiguator's majority-vote lemmas and tags are typically more accurate.

As an alternative approach to lemmatization, we tried fine-tuning a larger sequence-to-sequence model TinyBART[6] with `simpletransformers` (Rajapakse et al., 2024) on the UD_Belarusian-HSE corpus. TinyBART receives a combination of the word form and morphological tags produced by Stanza as input and generates the lemma as output. Earlier work on low-resource languages has shown a significant improvement in the performance of lemmatizers that use morphological tags as additional input (Afanasev and Lyashevskaya, 2023).

[6] https://huggingface.co/djulian13/be-tiny-bart

However, in a manually labeled random sample of 100 instances with TinyBART lemmas different from the majority-vote lemmas, the model doesn't seem to outperform Stanza (59: LLM is better; 4: TinyBART is better: 37: unclear). Both with Stanza and TinyBART, lemmatization errors indicate that the baseline model often struggles to generate the correct lemma where the disambiguator straightforwardly copies the only candidate lemma from input to output, that is, the lemmatization task might be inherently easier to solve with a two-stage pipeline.

4. Discussion and conclusion

We find that morphological disambiguation of Belarusian data with a strong LLM outperforms a neural baseline and produces results rather close to manual disambiguation (less so for tags than for lemmas, but the performance is strong given the difference in ambiguity rates). Running Belarusian texts through an LLM reveals negative transfer (from English, Russian, Ukrainian) and model-specific idiosyncrasies. Disambiguation consistency is higher for lemmas than tags. In small random samples, disambiguation quality appears to be generally higher for tags than lemmas, although the absolute number of errors in tags is larger, due to more massive ambiguity. When there is no list of candidate lemmas / tags to choose from, the LLM's predictions become less consistent and less accurate. Cases that are difficult for a human labeler to disambiguate are often difficult for an LLM as well. Conversely, wrong and doubtful LLM outputs highlight weak spots in morphological analysis that require more attention.

All data and scripts required to reproduce our results are available at https://doi.org/10.5281/zenodo.19349898.

5. Limitations and future work

The present study does not answer how large the performance gap is between open-weight and proprietary LLMs on the Belarusian morphological disambiguation task. The gold-standard evaluation in § 3.1 relies on a single 1K-word document, meaning the exact performance rankings are preliminary. The results of 100-instance random sample labeling can only be interpreted qualitatively, as the samples are too small for reliable statistical analysis. The Stanza baseline comparison in § 3.3 may be noisy because the tagset mapping is not 1-to-1.

Future work may include:

- utilizing structured generation to reduce the rate of failures and formatting errors in the LLM outputs;
- expanding the prompt to include more detailed labeling conventions and tricky case analysis;
- performing additional runs, as a way to further reduce the ratio of no-majority-vote instances;
- evaluating open-weight LLMs on the morphological disambiguation task;
- disambiguating the full YABC corpus and mapping the tags to the Universal Dependencies format, in order to facilitate training stronger tools for lemmatization and PoS tagging.

References

Ilia Afanasev and Olga Lyashevskaya. 2023. From web to dialects: how to enhance non-standard Russian lects lemmatisation? In *Proceedings of the 2023 CLASP Conference on Learning with Small Data (LSD)*, pages 167–175, Gothenburg, Sweden. Association for Computational Linguistics.

Savita Bhat and Vasudeva Varma. 2023. Large language models as annotators: A preliminary evaluation for annotating low-resource language content. In *Proceedings of the 4th Workshop on Evaluation and Comparison of NLP Systems*, pages 100–107, Bali, Indonesia. Association for Computational Linguistics.

Bosheng Ding, Chengwei Qin, Linlin Liu, Yew Ken Chia, Boyang Li, Shafiq Joty, and Lidong Bing. 2023. Is GPT-3 a good data annotator? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11173–11195, Toronto, Canada. Association for Computational Linguistics.

Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. 2023. ChatGPT outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences*, 120(30):e2305016120.

Google DeepMind. 2025. Gemini 3 Flash model card. Accessed: 2026-02-17.

Mika Hämäläinen. 2024. DAG: Dictionary-augmented generation for disambiguation of sentences in endangered Uralic languages using ChatGPT. In *Proceedings of the 9th International Workshop on Computational Linguistics for Uralic Languages*, pages 36–40, Helsinki, Finland. Association for Computational Linguistics.

Nataliia Kholodna, Sahib Julka, Mohammad Khodadadi, Muhammed Nurullah Gumus, and Michael Granitzer. 2024. LLMs in the loop: Leveraging large language model annotations for active learning in low-resource languages.

Anton Korikov, Pan Du, Scott Sanner, and Navid Rekabsaz. 2025. Batched self-consistency improves LLM relevance assessment and ranking. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 32687–32703, Suzhou, China. Association for Computational Linguistics.

Christopher D. Manning. 2011. Part-of-speech tagging from 97% to 100%: is it time for some linguistics? In *Proceedings of the 12th International Conference on Computational Linguistics and Intelligent Text Processing - Volume Part I*, CICLing'11, page 171–189, Berlin, Heidelberg. Springer-Verlag.

Cameron Morin and Matti Marttinen Larsson. 2025. Large corpora and large language models: a replicable method for automating grammatical annotation. *Linguistics Vanguard*, 11(1):501–510.

Omer Nahum, Nitay Calderon, Orgad Keller, Idan Szpektor, and Roi Reichart. 2025. Are LLMs better than reported? Detecting label errors and mitigating their effect on model performance. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 26782–26809, Suzhou, China. Association for Computational Linguistics.

Arbi Haza Nasution and Aytuğ Onan. 2024. ChatGPT label: Comparing the quality of human-generated and LLM-generated annotations in low-resource language NLP tasks. *IEEE Access*, 12:71876–71900.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online. Association for Computational Linguistics.

Thilina C. Rajapakse, Andrew Yates, and Maarten de Rijke. 2024. Simple transformers: Open-source for all. In *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, SIGIR-AP 2024, pages 209–215.

Yana Shishkina and Olga Lyashevskaya. 2021. Sculpting enhanced dependencies for Belarusian. In *Analysis of Images, Social Networks and Texts*, pages 137–147, Cham. Springer International Publishing.

Chahan Vidal-Gorène, Bastien Kindt, and Florian Cafiero. 2026. Under-resourced studies of under-resourced languages: lemmatization and POS-tagging with LLM annotators for historical Armenian, Georgian, Greek and Syriac. In *Proceedings of the Second Workshop on Language Models for Low-Resource Languages (LoResLM 2026)*, pages 324–334, Rabat, Morocco. Association for Computational Linguistics.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models.

Lars Wissler, Mohammed Almashraee, Dagmar Monett, and Adrian Paschke. 2014. The gold standard in corpus annotation. In *5th IEEE Germany Student Conference*.

Danni Yu, Luyang Li, Hang Su, and Matteo Fuoli. 2024. Assessing the potential of LLM-assisted annotation for corpus-based pragmatics and discourse analysis. *International Journal of Corpus Linguistics*, 29(4):534–561.

Judit Ács. 2025. *Morphology in the Age of Pretrained Language Models*. Ph.D. dissertation, Budapest University of Technology and Economics.

A. Prompt template

Consider the following grammatical tagset for Belarusian, presented here in Markdown-formatted tables:

[*The full tagset, one table per each PoS*]

Your input after a sequence of dashes will be a tokenized text in Belarusian with lemmas and grammatical tags conformant to the description above. The input has three tab-separated columns: token, all possible lemmas (pipe-delimited), all possible grammatical tags (pipe-delimited). The input isn't disambiguated, i.e. multiple lemmas and/or multiple tags are often suggested for a single token. When there are multiple candidate lemmas, the candidate tags are listed for the first lemma, then for the second lemma, etc.; in all other respects the order of candidates is arbitrary. Your task is to disambiguate the input. Requirements:

- Do not modify the token.
- Do not change the tokenization, even if it appears incorrect.
- If there are multiple candidate lemmas, choose the best fit. In rare exceptional cases it is possible that the correct lemma is not listed among the candidates; if you're 100% sure this is the case, you're allowed to suggest the lemma that you think is correct (please use this permission sparingly).
- If there are multiple candidate grammatical tags, choose the best fit. In rare exceptional cases it is possible that the correct tag is not listed among the candidates; if you're 100% sure this is the case, you're allowed to suggest the tag that you think is correct (please use this permission sparingly).
- If the tag is UNK, it means that the token wasn't recognized. In this case please suggest the appropriate lemma and tag.
- In all other cases, if there are no pipes in the input line, you may just copy it to the output without reviewing, as both the lemma and the tag will typically be correct. Again, you're permitted to edit them but only if you're 100% sure it is necessary.

Your output must have the same number of lines as the input, three tab-separated columns in each line, tokens copied from input to output, lemmas and tags disambiguated, i.e. no pipes and no UNK tags left.

For example, consider this input:

[*20 tokens with all candidate lemmas and tags*]

Then this could be the respective output:

[*20 tokens with disambiguated lemmas and tags*]

Now the actual input will follow below.

_____

[*Tokens with all candidate lemmas and tags*]