

Leveraging Large Language Models in Low-resourced Language NLP: A spaCy Implementation for Modern Tibetan

Yuki Kyogoku (Leipzig University), Franz Xaver Erhard (Leipzig University), James Engels (University of Edinburgh) and Robert Barnett (SOAS University of London)

Large Language Models (LLMs) are transforming the possibilities for developing Natural Language Processing (NLP) tools for low-resource languages. While languages like Modern Tibetan have historically faced significant challenges in computational linguistics due to limited digital resources and annotated datasets, LLMs offer a promising solution. This paper describes how we leveraged Google’s Gemini Pro 1.5 to generate training data for developing a basic spaCy language model for Modern Tibetan, focusing particularly on Part-of-Speech (POS) tagging. Combining traditional rule-based approaches with LLM-assisted data annotation, we demonstrate a novel methodology for creating NLP tools for languages with limited computational resources. Our findings contribute to the broader effort to enhance digital accessibility for low-resource languages while offering practical insights for similar projects in computational linguistics.

1 Introduction

Despite recent advancements in digital humanities, low-resource languages, such as Tibetan, still face substantial challenges due to limited digital resources and tools. Addressing these gaps is essential

for enhancing digital accessibility and supporting advanced linguistic research.

A key step in Tibetan language processing is the development of computational tools such as POS taggers, which underpin many NLP applications. However, creating such tools for Tibetan poses particular difficulties, including a lack of large-scale datasets, inconsistencies in existing text sources, and the distinct syntactic features of the language.

To address the scarcity of Tibetan-language corpora, a diverse dataset was compiled from contemporary sources, and we took additional steps to normalise text and correct inconsistencies such as punctuation errors and the use of abbreviations. Furthermore, integrating external tools, such as the Botok tokeniser (see section 3.1 below), is explored to manage structural challenges typical for the Tibetan language, particularly the absence of spaces between Tibetan words.

Finally, we demonstrate how using Google's LLM Gemini Pro 1.5 for automated data annotation critically contributed to developing a Tibetan language model for spaCy. Moreover, we showcased the potential of LLMs to contribute to the development of NLP resources more generally, aligning with the broader objective of addressing the digital divide for low-resource languages.

2 *The Scope of the Research*

The Divergent Discourses project¹ investigates the construction of narratives in Tibet in the 1950s and 1960s. It is interested in extracting

¹ The Divergent Discourses project received funding from the Deutsche Forschungsgemeinschaft (DFG) under project number 508232945 (<https://gepris.dfg.de/gepris/projekt/508232945?language=en>), and from the Arts and Humanities Research Council (AHRC) under project reference AH/X001504/1 (<https://gtr.ukri.org/projects?ref=AH%2FX001504%2F1>). For more information on Divergent Discourses, see <https://research.uni-leipzig.de/diverge/>. We also would like to thank Michael Richter and Tyler Neill for their valuable feedback on an earlier draft of this paper.

such narratives and related elementary information — such as agents, places, events, and their relationships — from digitised Tibetan newspapers. Although Automatic Text Recognition (ATR) is a critical component of our broader project (see Erhard 2025 in this issue), this paper focuses specifically on the process of training a basic spaCy language model for Modern Tibetan, using the digitised texts generated through ATR alongside digital-born materials.

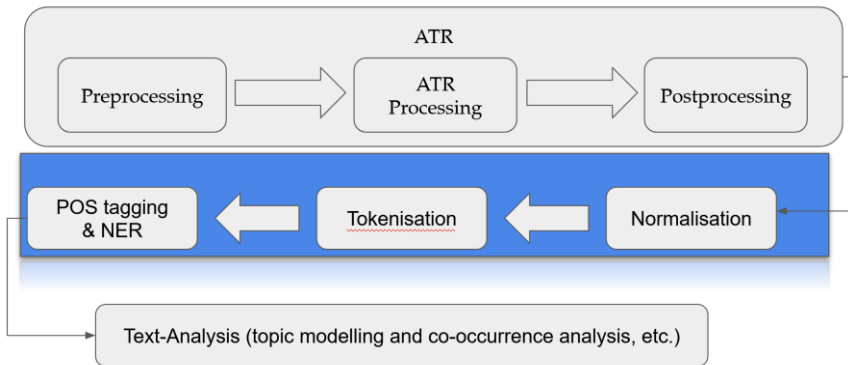


Figure 1 The complete project pipeline, with the scope addressed in this paper highlighted in blue. It is important to note that this figure should not be confused with the training process of a POS tagger.

The open-source natural language processing library spaCy is written in Python and Cython. It enables the training of pipelines for tasks such as POS-tagging, parsing, and Named Entity Recognition (NER) from scratch, even for languages not natively supported.² We chose to work with spaCy because it is the underlying NLP engine for a wide range of corpus mining and analysis tools, such as Wordless,³ or the integrated Leipzig Corpus Miner (iLCM),⁴ a tool offering various analytical functionalities such as topic modelling, co-occurrence analysis, and supervised text classification. The training process for the

² See <https://spacy.io/usage/spacy-101> (accessed January 25, 2025).

³ *Wordless* is an integrated corpus tool with multilingual support for studying language, literature, and translation developed by Ye Lei (叶磊), Shanghai. See <https://github.com/BLKSerene/Wordless> (accessed January 25, 2025).

⁴ <https://ilcm.informatik.uni-leipzig.de/> (accessed December 18, 2024), see also Niekler 2014, 2023.

spaCy model comprises two main phases: the generation of segmented and annotated training data and the subsequent training of the spaCy language model.

The structure of this study is organised as follows: First, the availability of NLP tools, such as tokenisers, POS taggers and related technologies, is discussed, with a focus on tools existing for the Tibetan language. This discussion also includes an overview of the authors' contributions, explicitly developing a Tibetan text normaliser and adapting the Botok tokeniser for the modern Tibetan language. Secondly, we examine the annotated corpora of Tibetan training data in general and emphasise the need to create a new Modern Tibetan corpus. Subsequently, we describe how an LLM such as Google's Gemini Pro 1.5 can be leveraged for the automatic creation of training data and thus overcoming the main obstacle of low-resourced languages: the lack of training data. The final section details the process of training a Modern Tibetan language model for spaCy.

3 *Relevant Tools for Tibetan*

This section discusses the techniques and digital tools relevant to the objectives of *Divergent Discourses* and examines existing digital tools for Tibetan. Despite recent advances in NLP, certain limitations persist in Tibetan digital humanities. These limitations are examined in detail below.

3.1 *Tokenisation*

Tokenisation is the process of breaking down a larger text into smaller units, called tokens, which can, depending on the context, be words, subwords, sentences, or even individual characters. However, in most cases, tokens correspond to what are commonly referred to as "words." Tokenisation is a fundamental step in NLP and computational linguistics and the first step in preparing text data for analysis or model training. Its quality directly impacts the accuracy of

subsequent steps, such as POS-tagging or NER. Thus, developing, selecting or customising an appropriate segmentation tool, known as a tokeniser, is a critical step in Tibetan language processing.

Unlike languages such as English, which use white spaces to separate words, Tibetan — often described incorrectly as monosyllabic — presents challenges for NLP because of its writing system and the absence of word delimiters. Therefore, a tokeniser is needed to divide strings of syllables (morphemes) into tokens.⁵

One notable example of a Tibetan tokeniser is Botok, developed by OpenPecha.⁶ Botok is designed as a rule-based tokeniser and was initially created to process Classical Tibetan texts. However, it is customisable, allowing users to adapt its functionality to specific text genres by altering its dictionary file. This adaptability makes Botok valuable for working with Tibetan texts in various contexts. The details of these customisations will be described in section 4.2 below.

3.2 POS-Tagging

In computational linguistics, a language model often needs to identify the grammatical roles of words in a text, such as nouns, adjectives, and other POS. This process, known as POS-tagging, is a critical step for many NLP tasks that require an understanding of syntactic distributions and permissible word combinations based on the grammatical features of a language. POS-tagging enhances the accuracy and efficiency of complex NLP tasks such as text parsing, machine translation, and text generation. In the context of Divergent Discourses, POS-tagging is essential because it facilitates the development of NER, which identifies words representing entities such as places, dates, people, or organisations. The pre-identification of proper nouns through POS-tagging simplifies the NER process.

⁵ For those who are interested in tokenisers, see https://huggingface.co/docs/transformers/v4.29.1/tokenizer_summary (accessed December 18, 2024).

⁶ <https://github.com/OpenPecha/Botok> and https://github.com/Esukhia/botok-data/tree/master/dialect_packs (accessed December 18, 2024).

While POS-tagging is not mandatory for all NLP tasks — topic modelling (see Schwartz & Barnett 2025 in this issue) and semantic search (see Engels & Barnett 2025 in this issue), for example, often rely on word embeddings instead, which do not need to have POS-tags — many widely-used NLP platforms, including spaCy and Stanford CoreNLP,⁷ require POS-tagging as part of the engine for newly trained language models, regardless of their specific architecture.

Tagging, however, is a computationally complex task. While some words consistently represent a single part of speech, others are homographic (e.g., “the sailor closed the hatches” vs. “the hen hatches an egg”) or polysemous across syntactic categories (e.g., “I hurt my back” vs. “the prime minister said he would back the new policy”). Addressing these challenges typically involves one of two approaches: (1) a rule-based system, which relies on predefined linguistic rules, or (2) a machine-learning-based system, which uses pre-annotated data to infer patterns. In the following subsections, we discuss various types of POS-taggers and introduce those specifically designed for Tibetan.

3.2.1 *The Rule-based Approach*

The rule-based approach to tagging, used by most early POS-taggers for any language (including the earliest taggers in English), seeks to replicate a human’s immediate linguistic intuitions by applying complex tree-based rulesets in which either specific morphologies or orthographic environments dictate a certain classification.

The earliest Tibetan taggers (and many taggers still in use by the few Tibetan NLP researchers and users) were rule-based, using computational versions of older syntactic descriptions of Tibetan. This is the case with Hackett (2000), which applies Wilson’s (1998) general rules of Tibetan syntax. This tagger operates essentially like a decision tree: locate the word > check the word in the dictionary > decide which

⁷ <https://stanfordnlp.github.io/> (accessed July 12, 2024).

POS is most likely given the immediately preceding or neighbouring POS > assign the relevant tag.⁸

The Buddhist Digital Resource Centre (BDRC) and Esukhia developed a more complex form of rule-based tagging using a modified rule-based approach. Botok, in addition to tokenisation, also applies POS tags to its tokens by looking for those tokens in an internal dictionary and returning the most likely result based on the morphological features of the word and its neighbours. The following diagram describes the general architecture of Botok's tagger:

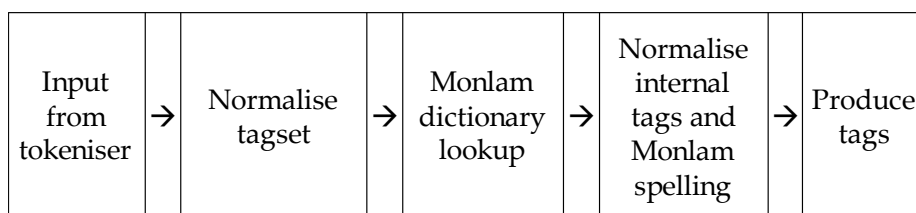


Figure 2 General architecture of Botok's tagger.

This tagger, however, does not allow to any significant extent for conditional probabilities based on the syntactic environment of the token it tries to tag. Most of the time, the tag defaults to the first listed POS in Monlam's online dictionary.⁹ To measure the overall quality of Botok's POS tagger, we compared its POS tags on a small sample text to those of an expert consultant who is both a Tibetologist and a theoretical linguist (see Appendix A).¹⁰ We compared each tag to the expert's gold-standard tag set, using a sliding window of two tokens to accommodate splitting decisions involving syllables where even experts disagree. In a 30-word sample paragraph, compared to the expert reviewer, Botok's POS-tagger performed to a middling degree,

⁸ Detailed information about the internal function of the tagger is not included in the original publication, and the source code is not easily retrievable online.

⁹ <https://monlamdictionary.com/> (accessed December 18, 2024).

¹⁰ We thank Dr Camille Simon (Paris) for her assistance with this process.

with an agreement accuracy of 53.3% (16/30).¹¹ Botok's tagging facility is thus relatively elementary compared to its tokenising capability.¹²

Other rule-based tagging systems exist for Tibetan. A system for syntactical analysis called Constraint Grammar, for example, has been successfully built into Tibetan taggers (see Faggionato & Garrett 2019, Garrett & Hill 2015), yielding better results than a simple dictionary-lookup system such as the one included with Botok. However, systems based on Constraint Grammar lack the learning flexibility that one would find in a machine-learning model.

Rule-based taggers are, in general, limited by a few critical weaknesses independent of resource availability:

- (1) Unless combined with adaptive conditional probabilities, rule-based taggers are entirely context-independent. Thus, when a tagger of this type encounters an unfamiliar syntactical structure, it is liable to make an incorrect decision or to apply a tag such as "other," "unknown," or "none."
- (2) A subset of the context-independence problem is the problem of polysemy: how should the parser decide to tag a word that can, at different times, function as a noun, verb, adjective, or something more fine-grained such as a past participle?
- (3) Unstructured text, or text in unfamiliar genres, can confuse rule-based taggers that are overfitted to certain textual environments or contexts. For example, suppose the rules on which a tagger is

¹¹ This score was achieved with the aid of a sliding window (since Botok tends to be "splitty", i.e., it tokenises longer compounds in its elementary syllables) and after allowing for Botok's consistent quasi-errors (almost everything Botok considers "other" is a noun, for instance). In addition, Botok's tagset is more limited than the Universal Dependencies (UD) tagset we instructed our annotator to use, and even for humans, it is arguable how to assign POS-tags in the UD tagset to such Tibetan linguistic objects as the genitive case ending, and we therefore give Botok some leeway, such as when it assigned PART to tokens our annotator tagged as AUX or ADP.

¹² It should be noted that Botok was developed with tokenisation in mind, and its POS-tagging capabilities are more of a side effect, according to one of the developers (Personal conversation with Hélios Drupchen Hildt, Diverge project meeting SOAS London, 26 August 2024).

based have been developed based on the conventions found in literary or religious texts. In that case, it will have difficulty tagging newspaper text. Social media, for example, a prominent use-case for NLP applications, poses a significant challenge to rule-based taggers because of the unending linguistic creativity of the public in informal linguistic contexts.

- (4) If dialectal variation is a concern, such as if the target language is pluricentric (Hindi-Urdu, Serbo-Croatian), and particularly when that variation consists of minor differences in morphology and syntax, these differences must be explicitly compensated for in the ruleset.

Prior to the development of LLMs and other transformer-based systems, rule-based taggers were generally the only tools available for Tibetan, despite its millions of speakers and a relative abundance of online resources. In addition, the amount of readily available annotated training data in Tibetan is unusually low compared to other languages of its size and reach.

3.2.2 *Machine-learning Approaches to Tagging*

There has nevertheless been an evolution in POS-tagging for Tibetan from a rule-based approach to an approach based on “shallow” machine learning. This was demonstrated in the case of Tibetan by the ACTib tagger developed by Meelen and Hill (2017), which combines a rule-based with a memory-based tagger.

ACTib performs word and sentence segmentation and provides POS-tagging, offering a high level of detail. It is designed explicitly for Classical Tibetan. The combination of a rule-based approach with the Tilburg Memory-Based Tagger (van der Sloot & van Gompel 2024) allows for more flexibility and yields better results for a wide range of texts than a simple rule-based system (Meelen 2021). ACTib has trained the Tilburg Memory-Based Tagger for use with Classical Tibetan, but it could be retrained for use with modern Tibetan. However, due to a lack of appropriate training data, it is impossible to

retrain the memory-based tagger of ACTib for our specific needs. Consequently, ACTib and its output did not meet the requirements of our study.

3.2.3 *Transformer Approaches to Tagging*

The shallow machine-learning approach may now become outdated by the major improvements represented by the development of transformers and transformer-based approaches to NLP tasks.¹³ Tibetan LLM technology, such as the projects underway at Monlam AI,¹⁴ are rapidly improving and will sooner or later gain an “intuitive” understanding of Tibetan.¹⁵ GPT-4 is already capable of intuitively POS-tagging text from medium-resource languages like Vietnamese with nearly 100% accuracy. By September 2024, it could POS-tag a Tibetan sentence with around 80-90% accuracy. A comparison of POS-tagging in Tibetan by GPT-4, Claude 3.5 Sonnet, and Gemini shows (besides several allowable disagreements, mostly resulting from variations in their approach to tokenising and from variations in their tag sets) a similar error rate of 10-20%, which is low given the likely rate of progress (see Appendix F).

However, all LLMs are black boxes, and their output is difficult to predict. They hallucinate, occasionally add irrelevant information and generally share a tendency to be inconsistent, i.e., sporadically making differing tagging or translation decisions. We found that if the exact

¹³ A transformer is a type of deep learning model widely used in tasks like language translation, text generation, and summarisation. Unlike earlier models, which process text sequentially (word by word) and apply limited contextual awareness to decision-making, transformers use an attention mechanism that allows them to focus on the most relevant parts of a sentence or sequence. This helps them understand the relationships between words more effectively, capturing context and meaning even across long distances within the text.

¹⁴ <https://monlam.ai/about> (accessed December 18, 2024), and <https://github.com/MonlamAI> (accessed December 18, 2024).

¹⁵ T-LLaMA, a first fine-tuned LLM for Tibetan based on META's LLaMA2, was published in December 2024, but could not yet be tested by Divergent Discourses, see Lv *et al.* 2025.

text is repeatedly submitted for tokenisation, the results are not always identical, as would be expected. Consequently, to leverage LLMs' intuitive knowledge of Tibetan, an equally powerful mechanism needs to be installed that can control and minimise inconsistencies as well as the rare yet unavoidable hallucinations so characteristic of LLMs.

Several dedicated POS taggers have been developed for Tibetan using Deep Learning and Transformer-based approaches. Li *et al.* (2022) combine deep learning techniques, specifically Bidirectional Long Short-Term Memory (Bi-LSTM) and Iterated Dilated Convolutional Neural Network (IDCNN), with machine learning methods, such as Conditional Random Fields (CRF), to propose an end-to-end model for joint Tibetan word segmentation and POS-tagging. Similarly, Xiangxiu *et al.* (2022) employ Embeddings from Language Models (ELMo) and the self-attention mechanism of Transformers to address challenges related to polysemy and out-of-vocabulary words. However, while the results of these Transformer-based POS-taggers are promising, none of the tools is publicly available, preventing their practical use.¹⁶

3.3 Normalisation of Texts

Normalisation is the process of reducing the randomness of a text such as various encodings, unnecessary characters to a set standard. This can also include more complex tasks such as stemming stripping, i.e., eliminating affixes, or lemmatisation, i.e., reducing variants to a base form. Tibetan, unlike languages such as Russian or Sanskrit, is an agglutinative language. This allows for direct searches of specific words, or at least of their stems, except in the case of conjugated verbs. However, several factors can complicate direct search functionality. For example, suppose traditional Tibetan brackets or numerals are found in the text alongside their non-Tibetan counterparts. In that case,

¹⁶ BERT models have been developed by Tibet University and made available on Hugging Face (<https://huggingface.co/UTibetNLP>, accessed December 18, 2024). However, they are not specifically designed for POS-tagging.

these will be missed by a search for the standard, non-Tibetan version. The *tsheg* (།), when — as is common practice — used repeatedly as a filler at line breaks, may break up a search string and make a search unfeasible. Abbreviations, prevalent in early 20th-century Tibetan publications such as the *Tibet Mirror*, can also confuse a standard search engine. Only with the replacement of lead typesetting in the late 20th century with computerised typesetting did abbreviations become less frequent. To address these issues, the project developed a normaliser that converts such features into modern, standardised forms (Kyogoku *et al.* 2024a). It replaces traditional brackets and numerals with standard modern ones, e.g., the traditional Chinese quotation mark (「...」) is replaced by (“...”). It resolves simple abbreviations, such as the common contraction of final -ལས by -ལ, but also resolves complex abbreviations, such as འདྲེ་ into འདྲེ་ལས་ལས་ by referring to a list extracted from our newspaper corpus and a revised version of the list of more than 6,500 abbreviations compiled by Bruno Laine.¹⁷ The normaliser also corrects errors such as improper punctuation. For instance, if a *tsheg* is missing between the letter *nga* (ཨ) and the punctuation mark *shad* (།), the normaliser automatically inserts a *tsheg* to ensure text consistency. These normalisations do not alter the raw or original text in the corpus but are saved in a new version that allows a user to retrieve a more complete set of results from a single search.

4 Corpus of Training Data for Modern Tibetan

A text corpus of training data is a digitised collection of language data with specific linguistic annotation designed for computational analysis. The CoNLL-U format is often employed in linguistic research among the various formatting standards for such corpora. This format, commonly used in Universal Dependencies (UD), presents annotated sentences with POS tags and dependency labels, etc., in the CoNLL-U

¹⁷ <http://www.rkts.org/abb/index.php> (accessed December 18, 2024).

format.¹⁸ UD sets of POS tags have been developed for many languages, including classical ones. In addition, platforms like spaCy require their training data to be formatted in CoNLL-U. As a result, the CoNLL-U format has become the standard for syntactic and morphological annotation. We, therefore, formatted our training data in the CoNLL-U format to ensure compatibility for subsequent training of our spaCy Tibetan language model.

As highlighted in the introduction, Tibetan, like many low-resource languages, so far has limited amounts of data available for use in training. In addition, a UD set has not yet been developed for either Modern or Classical Tibetan. The requirements and structure for a Classical Tibetan treebank were outlined by Faggionato and Meelen (2019), but such a resource – again – would have limited applicability to our mid-20th-century Tibetan texts. Although some CoNLL-U-formatted files exist for Classical (Faggionato *et al.* 2021) and Modern Tibetan (Dakpa *et al.* 2021), their quality and volume are insufficient for training an effective POS tagger.

4.1 Automatic generation of a training dataset for modern Tibetan

For the reasons outlined in the previous section, we decided to create our own training dataset so that we would then be able to train a spaCy model for modern Tibetan.

The initial step in creating a CoNLL-U file with Tibetan POS tags was the collection of a raw text corpus sufficiently large enough to train a spaCy language model. Practical constraints limited the corpus size to 100–200 MB. Data sources included Ground Truth transcriptions of Tibetan newspapers from the 1950s and 1960s, book publications from the same period, contemporary Tibetan newspapers from South Asia, and scraped content from openly accessible Tibetan news websites, particularly *Tibet Daily* (Kyogoku *et al.* 2024b).

Prior to the widespread use of LLMs, training datasets had to be manually annotated by human annotators. However, by mid-2024

¹⁸ <https://universaldependencies.org/format.html> (accessed December 18, 2024).

LLMs such as ChatGPT, Gemini, or, more recently, Claude 3.5 Sonnet (released June 20, 2024), although not specifically trained on Tibetan material, were showing varying yet significant degrees of accuracy in their uses of Tibetan language. Since using LLMs as annotators for low-resource languages can save considerable time and effort compared to traditional manual methods, we opted to use LLMs to generate the necessary annotations automatically. We evaluated three LLMs, including ChatGPT, and identified Gemini Pro 1.5 (released February 15, 2024)¹⁹ for its – at the time – superior accuracy and accessibility in Google's cloud environment as the best option for POS tagging Tibetan sentences (Appendix F and Barnett & Engels 2025: 25-28, 36-38 in this issue). To generate an annotated dataset appropriate for spaCy training, we implemented several optimisations in the prompt, i.e., the instruction or text provided to a LLM to elicit a specific response:²⁰

- (1) **Entry selection:** We limited the automatic annotation to essential fields in the prompt: ID, form, and UPOS (Universal Part-of-Speech Tag), as a complete CoNLL-U format was unnecessary for our objectives.
- (2) **English translation:** Recent studies, such as Huang (2023), have demonstrated that incorporating English translations of non-English languages, particularly those classified as low-resourced, can significantly enhance performance in tasks related to language understanding, reasoning, and generation. Building on this finding, alongside Tibetan text, requests for English translations are included in the prompt to provide additional context, in order to improve the accuracy of the linguistic annotations.
- (3) **Pre-segmentation of sentences:** We found inconsistencies in Gemini's word segmentation of modern Tibetan texts. We

¹⁹ <https://ai.google.dev/gemini-api/docs/models/gemini> (accessed December 18, 2024) and <https://blog.google/technology/ai/google-gemini-next-generation-model-february-2024/#performance> (accessed December 18, 2024).

²⁰ There are other useful techniques for improving the LLM performance. See Chen et al. (2023) and Deshpande et al. (2024).

therefore used Botok to segment our data prior to inputting it into the model, ensuring that token boundaries were preserved as needed. For this study, a customised “dialect pack” of Botok was used (see the detailed explanation in section 4.2. below).

- (4) **Structured prompt specification:** We specified a CoNLL-U-compatible format within the prompt, with entries separated by tabs, and extracted only the tab-separated line. This ensured that the output contained solely the table without any additional notes occasionally appended by Gemini. For instance, despite the differences in the output in Figures 3 and 4, only the tab-separated entries, i.e., the entries in the table (token number, token, English translation, and POS-tag) were extracted and saved in CoNLL-U format; additional content was disregarded.

Response

1	གངས་འབྲུགས་	snow_and_ice	NOUN
2	ལུས་རྩལ་	sports	NOUN
3	པས་	by	ADP
4	འགྲན་རར་	competition	NOUN
5	མོས་ཚོག་པའི་	can_show	VERB
6	གྲུབ་འབྲས་	achievement	NOUN
7	གཏོད་པ་དང་	achieved	VERB

Figure 3 A response generated by Gemini Pro 1.5, presented in a table format without any additional annotations.

- (5) **Limiting the Prompt Size:** An increase in the length of the prompt can cause interruptions in Gemini’s answer generation. To address this, we had to ensure a reasonable prompt length and segmented the input Tibetan sentence using the *shad* (།) as a delimiter (note: this delimiter does not always signify the end of a sentence; see Figure 5).

CoNLL-U format for the Tibetan passage:

Token Number	Token	English Translation	POS-Tag
1	གངས་འབྲུག་གས་	glacier	NOUN
2	ལུས་རྒྱལ་པ་	athletes	NOUN
3	ས་	by	ADP
4	འབྲུག་རྒྱལ་	competition	NOUN
5	འདི་	in	ADP
6	ལྟོ་སྟོན་པ་	show, demonstrate	VERB
7	འདི་	that can be	PART
8	ལྷན་འབྲེས་	results	NOUN
9	མཐོང་པ་	achieved	VERB
10	དང་	and	CCONJ

Notes:

- Some POS tags might be debatable depending on the context and interpretation.
- The English translations are approximate and provided for understanding, not as a one-to-one mapping.
- The CoNLL-U format typically includes more columns (lemma, morphological features, etc.), but only the requested ones are provided here.

Figure 4 Response with additional annotations generated by Gemini Pro 1.5

- (6) **Correction of Annotation Inconsistencies (Post-Processing):** The results produced several inconsistencies, such as the genitive suffix or particle, which was variably labelled as ADP or PART.²¹ We addressed these discrepancies by applying standardisation rules in the post-processing stage, using a Python script. Additionally, if a single token in a sentence contained an unexpected POS-tag, not conforming to the UD standard, the whole sentence was eliminated by the script from the dataset. Consequently, the volume of data Gemini generates

²¹ As for the POS-tags, see <https://universaldependencies.org/u/pos/index.html> (accessed January 15, 2025).

is smaller than the initial input dataset, albeit with a higher POS-tag accuracy.²²

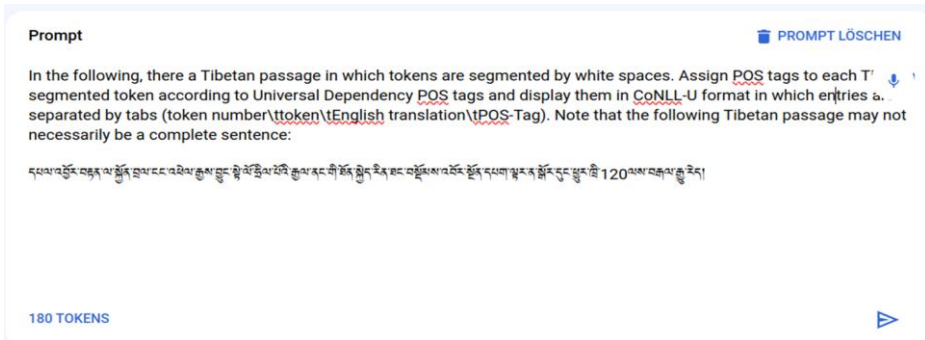


Figure 5 Example of a prompt. Note that this prompt was manually created in a browser, while our approach involves automatically generating prompts.

We found that in general, one-shot or few-shot prompts — where one or a few examples of the desired output are provided in the prompt — tends to perform better than zero-shot prompting, which does not include examples (Sivarajkumar 2023). However, this approach was not feasible because of the limitations of the prompt’s length (see point 5 in the paragraph above). Still, the evaluation of the training dataset produced by Gemini Pro 1.5 yielded promising results. A sample of 100 sentences was randomly extracted and evaluated by an expert in our team: 93% of the tokenisations and 91% of the POS-tags were deemed to be accurate (see Appendix B).

²² The spaCy tutorial page (<https://spacy.io/usage/training>, accessed January 15, 2025) states, “If you want to train a model from scratch, you usually need at least a few hundred examples for both training and evaluation” The generated dataset (Kyogoku *et al.* 2024b) comprises approximately 50 MB of data (training data 39.3 MB + validation data 9.6 MB), and the dataset contains over 80,000 sentences (training data 64927 sentences + validation data 16231 sentences). The average number of tokens per sentence is approximately 7.5 (training data 7.41 + validation data 7.25 tokens).

4.2 Customisation of Botok

When we first reviewed the results, we noticed that Gemini's tokenisation decisions, while generally acceptable, lacked consistency in some cases. However, consistent data is essential for training a spaCy language model. Consequently, as outlined in the third listed optimisation, we opted to enforce consistent tokenisation on Gemini by providing it with text pre-tokenised with Botok.

To enable support for Modern Tibetan, we used Botok's functionality, which allows users to adapt it to more specific language varieties by including "dialect-packs," which incorporate purpose-specific dictionaries. We created a "dialect-pack" for Modern Tibetan by modifying Botok's dictionary file (*tsikchen.tsv*).²³ We merged seven Modern Tibetan dictionaries with the original dictionary provided by Botok.²⁴ Duplicate entries were removed during this process.

However, the initial version of the resulting dictionary contained numerous compounds consisting of multiple words. Since our approach prioritises tokenising words into their smallest possible units, we removed compounds formed by genitive particles or those where adjectives or numbers modified nouns. Additionally, we limited the maximum number of syllables per word to four, as most Tibetan words fall within this range. This customised dictionary enabled tokenisation that aligns more effectively with our research objectives (Erhard *et al.* 2024).²⁵

²³ As for how to customise dialect packs, see <https://github.com/OpenPecha/Botok?tab=readme-ov-file> (accessed January 15, 2025).

²⁴ The custom dictionary was compiled from Christian Steinert's collection (<https://github.com/christiansteinert/tibetan-dictionary/tree/master/input/dictionaries/public>, accessed January 12, 2024) and contains the following dictionaries: Grand Monlam Dictionary (default dictionary of Botok), [Jim Valby](#), [Ives Waldo](#), [Dan Martin](#), [Tshig mdzod chen mo](#), [Dung dkar](#), and [Tibetan Terminology Project](#). The resulting dictionary was cleaned up and edited to the project's requirements by removing double entries, phraseologisms, ungrammatical entries, etc. Moreover, ca. 1000 personal and place names compiled from the project's material were added; see Erhard and Xiaoying 2025 in this issue.

²⁵ The evaluation of the customised Botok, referred to as Modern Botok, is presented in Table 1 in Section 5.

4.3 Challenges in the generation of the training dataset

Some significant challenges remain unresolved in the process of generating a training dataset using Gemini Pro 1.5. One issue was the incorrect recognition of proper nouns foreign to the Tibetan language, as illustrated in Figure 6. This problem is mainly limited to proper nouns introduced to Tibetan through Chinese, particularly those which imitate Chinese pronunciation.²⁶ Most of these names are originally Chinese and consist of three syllables, as in the example of ལི་ཀེ་ཁྲོང་ (Li Keqiang 李克强), the former Chinese premier (2013–2023). A potential solution involves incorporating proper names into the dictionary file of the Modern Botok dialect-pack (tsikchen.tsv); however, this has not yet been fully implemented.²⁷

```
## sent_id = 1
# text = ལི་ཀེ་ཁྲོང་ ལི་ཀེ་ཁྲོང་ ལི་ཀེ་ཁྲོང་ ལི་ཀེ་ཁྲོང་
1      ལི་      -      PROPN  -      -      -      -      -
2      ཀེ་      -      PROPN  -      -      -      -      -
3      ཁྲོང་   -      PROPN  -      -      -      -      -
4      ལི་      ལི་      ADP    -      -      -      -      -
5      ལི་ཀེ་ ལི་ཀེ་ ལི་ཀེ་  VERB  -      -      -      -      -
6      -      -      PUNCT  -      -      -      -      -
```

Figure 6 Li Keqiang ལི་ཀེ་ཁྲོང་, the name of the Chinese premier (2013–2023), is incorrectly tokenised into individual components.

5 Training a Basic spaCy Model for Modern Tibetan

This section describes the development of a modern Tibetan spaCy language model with the automatically created training data described in the previous section, comprising a pipeline that includes a POS-tagger, for integration into the iLCM. Given that the iLCM uses spaCy

²⁶ For the many ways in which foreign personal names and toponyms are rendered in Tibetan, see Erhard & Xiaoying 2025.

²⁷ In the current version of Modern Botok, we included more than 1,200 personal and place names found in newspapers from the 1950s and 1960s (Erhard *et al.* 2024). However, this did not include the majority of Chinese or foreign names found in the later newspapers that we included in the dataset.

version 3.2.6, we conducted training with version 3.2.1 to ensure compatibility.²⁸

As an initial experiment, a spaCy language model, Tibetan for spaCy 1.1, was trained using texts in which Tibetan sentences were artificially segmented by white spaces with Botok, which had not yet been customised as described above, thereby avoiding directly integrating Botok into spaCy (Engels *et al.* 2023). It is important to note that during this training process, the model treated input sentences as if they were in the English language, with the language parameter in the configuration file set to English (`lang = 'en'`). Although this model enabled Tibetan language support on iLCM, performing tokenisation with the same accuracy as Botok, it has no POS-tagging capability, a prerequisite for many downstream NLP tasks.

5.1 *Training of the Modern Tibetan spaCy model*

In contrast, the approach discussed in this section involved training a spaCy model from scratch, i.e., setting the language parameter in the configuration file to multi-language (`lang = 'xx'`) and integrating Botok into the training process.

However, relying solely on Botok proved insufficient for developing a functional spaCy model for modern Tibetan. Although returning no error messages during training, the resulting spaCy model often misassigned POS-tags in an apparently random manner. This indicated the need to adjust the spaCy model's training configuration further.

Only through extensive experimentation we identified two critical configuration modifications that were necessary to handle the Tibetan data: (1) setting the pipeline to [`"tok2vec", "morphologizer"`] instead of [`"tok2vec", "tagger"`], and (2) adding the phrase `"SpaceAfter=No"`

²⁸ Note that the iLCM requires spaCy version 3.2.6, while the newest version was 3.8.2 at the time of writing. As the computational demands of model training exceeded the capabilities of a personal laptop, the models for Divergent Discourses were trained on the high-performance computing cluster maintained by the Scientific Computing team at Leipzig University (<https://www.sc.uni-leipzig.de/>).

in the MISC column of the CoNLL-U file.²⁹ We adopted this approach based on observed conventions in other language-specific CoNLL-U files, while additional settings could possibly further enhance the model’s performance.

```

# sent_id = 1
# text = ། ལྟེ་མེ་
1      །          །          PUNCT  _      _      0      root  _      SpaceAfter=No
2      ལྟེ་མེ་   ལྟེ་མེ་   NOUN   _      _      0      root  _      SpaceAfter=No

# sent_id = 2
# text = རྟེ་ཡང་ ལྷ་ཚད་
1      རྟེ་ཡང་   རྟེ་ཡང་   ADV    _      _      0      root  _      SpaceAfter=No
2      ལྷ་ཚད་   ལྷ་ཚད་   NOUN   _      _      0      root  _      SpaceAfter=No

# sent_id = 3
# text = ལྷན་ ལྷ་ འགའ་ ལྷ་ དང་ །
1      ལྷན་       ལྷན་       NOUN   _      _      0      root  _      SpaceAfter=No
2      ལྷ་         ལྷ་         ADP    _      _      0      root  _      SpaceAfter=No
3      འགའ་ ལྷ་   འགའ་ ལྷ་   NOUN   _      _      0      root  _      SpaceAfter=No
4      དང་       དང་       CCONJ  _      _      0      root  _      SpaceAfter=No
5      །         །         PUNCT  _      _      0      root  _      SpaceAfter=No

# sent_id = 4
# text = འགྲ་ ལྷམ་ འདྲེ་ ལྷགས།
1      འགྲ་ ལྷམ་   འགྲ་ ལྷམ་   NOUN   _      _      0      root  _      SpaceAfter=No
2      འདྲེ་ ལྷགས། འདྲེ་ ལྷགས། NOUN   _      _      0      root  _      SpaceAfter=No
3      །         །         PUNCT  _      _      0      root  _      SpaceAfter=No

# sent_id = 5
# text = མངས་ ལྷམ་ འཛོམ་ ལྷན་ འདམ་ ལྷམ་ འཛི་ འགྲན་ ལ་ ལྷང་ ལྷམ་ ལྷང་ ལྷ་ དར་ ཡོང་ ལྷམ་ ལ་ ལྷག་ ལྷངས་ ཡོད།
1      མངས་ ལྷམ་   མངས་ ལྷམ་   NOUN   _      _      0      root  _      SpaceAfter=No
2      འཛོམ་ ལྷན་ འཛོམ་ ལྷན་   NOUN   _      NOUN   0      root  _      SpaceAfter=No
3      འཛི་       འཛི་       ADP    _      _      0      root  _      SpaceAfter=No
4      འགྲན་ ལ་   འགྲན་ ལ་   PRON   _      _      0      root  _      SpaceAfter=No
5      ལྷང་       ལྷང་       PRON   _      _      0      root  _      SpaceAfter=No
6      འགྲན་ ལ་   འགྲན་ ལ་   NOUN   _      _      0      root  _      SpaceAfter=No
7      ལྷང་       ལྷང་       NOUN   _      _      0      root  _      SpaceAfter=No
8      ལྷམ་       ལྷམ་       ADP    _      _      0      root  _      SpaceAfter=No
9      ལྷང་       ལྷང་       NOUN   _      _      0      root  _      SpaceAfter=No
10     ལྷ་         ལྷ་         ADP    _      _      0      root  _      SpaceAfter=No
11     དར་       དར་       VERB   _      _      0      root  _      SpaceAfter=No
12     ཡོང་       ཡོང་       VERB   _      _      0      root  _      SpaceAfter=No
13     ལྷམ་ ལ་   ལྷམ་ ལ་   PRON   _      _      0      root  _      SpaceAfter=No
14     ལྷག་       ལྷག་       DET    _      _      0      root  _      SpaceAfter=No
15     ལྷངས་       ལྷངས་       VERB   _      _      0      root  _      SpaceAfter=No
16     ཡོད།       ཡོད།       AUX    _      _      0      root  _      SpaceAfter=No
17     །         །         PUNCT  _      _      0      root  _      SpaceAfter=No

"train.conllu" 675979L, 39335983B

```

Figure 7 The final version of the CoNLL-U file produced by Gemini Pro 1.5.

Additionally, as the iLCM requires either a sentencizer or dependency parser to recognise sentence boundaries,³⁰ a post-pro-

²⁹ For a detailed explanation, refer to our discussion in the spaCy forum: <https://github.com/explosion/spaCy/discussions/13549> (accessed January 15, 2025).

³⁰ This is mainly to avoid errors such as: “ValueError: [E030] Sentence boundaries unset. You can add the 'sentencizer' component to the pipeline with:

cessing step was implemented. In this step, the HEAD and DEPREL fields in the CoNLL-U file were automatically populated with the placeholder values “root” and “0” respectively, allowing the training of a parser to function as a sentencizer. Despite such challenges during training with spaCy, we successfully developed a basic spaCy model for modern Tibetan (Kyogoku *et al.* 2024c).

5.2 Evaluation of the Modern Tibetan spaCy model

The accuracy scores (correct predictions / all predictions) of the Modern Tibetan spaCy, compared with the experimental Tibetan for spaCy 1.1 (based on an English language model) and Modern Botok, i.e., the customised Botok, are as presented in Table 1.³¹

Table 1 Evaluation table for tokenisation and POS-tagging.

<i>Method</i>	Tokenisation	POS-tagging
<i>Tibetan for spaCy 1.1</i>	0.931 (1500/1611)	N/A
<i>Modern Botok</i>	0.957 (1510/1578)	N/A
<i>Modern Tibetan spaCy</i>	0.924 (1394/1509)	0.872 (1316/1509)

All three methods demonstrate strong performance in terms of tokenisation. Tibetan for spaCy 1.1, which requires input texts with words separated by white spaces, tokenises texts precisely as the default, uncustomised Botok (see Appendix C). Notably, the tokenisation scores for both Modern Botok (see Appendix D) and

``nlp.add_pipe('sentencizer')`. Alternatively, add the dependency parser or sentence recogniser or set sentence boundaries by setting `doc[i].is_sent_start`."`

³¹ In our evaluation standard, the tokenisation of compounds, such as ལྷོ་ལོ་ལོ་ ("outside and inside") and ལྷོ་ལོ་ལོ་ ("biggest"), into their constituent parts — ལྷོ་ ("outside") and ལོ་ ("inside"), or ལྷོ་ ("big") and ལོ་ ("most") — is permissible, alongside their undivided forms. On the other hand, if personal or place names are fragmented into meaningless components, such as ལྷོ་ལོ་ལོ་ (Xi Jinping 习近平) being split into ལྷོ་, ལོ་, and ལོ་, all syllables constituting the compound are considered erroneous in the evaluation. This evaluation uses the same dataset as described in Appendix B, comprising of 100 randomly selected sentences (see also Section 4.1).

Modern Tibetan spaCy (see Appendix E) differ. At the same time, they should be identical, in theory, given that the tokenisation in both cases is based on Botok. Still, the score for Modern Botok is slightly higher than that of the Modern Tibetan spaCy.

The relatively weaker performance of Modern Tibetan spaCy points to a limitation associated with using spaCy to process Tibetan language. First, since Tibetan texts have no word segmentation, spaCy requires the integration of the external tokeniser Botok, even though tokenised training data is provided in the CoNLL-U file. Since the CoNLL-U file generated by Gemini Pro 1.5 reflects the tokenisation results of Botok, presumably, the weakness is introduced in spaCy's training process.³² Nevertheless, it is worth highlighting the high POS-tagging score achieved by Modern Tibetan spaCy.³³

6 Conclusion

By addressing the unique syntactic features of Tibetan and overcoming the challenges posed by the scarcity of annotated corpora of modern Tibetan texts, a Modern Tibetan spaCy language model could be trained. This study demonstrated the potential of LLMs such as Gemini Pro to generate training data for low-resourced languages automatically. Divergent Discourses accordingly produced an automatically POS-tagged corpus of modern Tibetan within a relatively short time, facilitating the successful training of a spaCy language model. Existing bugs will likely be fixed by either improving Botok's tokenisation (e.g., by employing a fine-tuned Tibetan LLM) or

³² Notably, this does not apply to our experimental Tibetan for spaCy 1.1. model, which was trained on Tibetan text segmented by white spaces.

³³ POS-tags associated with tokens evaluated as correct are assessed accordingly. Conversely, POS tags associated with tokens identified as erroneous are generally considered erroneous. As a result, the accuracy score for POS-tagging is inherently lower than that for tokenisation. When the number of correct predictions is divided by the total number of tokens evaluated as correct, the resulting accuracy score is 0.976 (1316/1394).

improving the POS tagging capabilities of LLMs such as Gemini, Claude, or ChatGPT.

The current Modern Tibetan spaCy model allows researchers to perform NLP analysis of Tibetan language materials with corpus analysis tools dependent on spaCy. The availability of a Tibetan language model for spaCy, an industry standard and openly accessible NLP platform, represents a significant step forward in enhancing digital accessibility and advancing linguistic research on Tibetan language textual sources. Although this research is an ongoing effort and future work will focus on enhancing and fine-tuning, particularly on enabling NER functionality for the Modern Tibetan spaCy model, the authors hope that the presented approach and workflow can function as a starting point for similar research on other under- and low-resourced languages.

Bibliography

Barnett, Robert, Nathan W. Hill, Hildegard Diemberger, and Tsering Samdrup

"Named-Entity Recognition for Modern Tibetan Newspapers: Tagset, Guidelines and Training Data," *Zenodo*, 2021. [doi:10.5281/zenodo.4536516](https://doi.org/10.5281/zenodo.4536516).

Engels, James, Franz Xaver Erhard, Robert Barnett, and Nathan W. Hill
"Tibetan for Spacy 1.1," *Zenodo*, 2023. [doi:10.5281/zenodo.10120779](https://doi.org/10.5281/zenodo.10120779)

Erhard, Franz Xaver

"Text and Layout Recognition for Tibetan Newspapers with Transkribus," *Revue d'Etudes Tibétaines* (74), 2025, pp. 128–171.

Erhard, Franz Xaver, Yuki Kyogoku, Robert Barnett, and Nathan W. Hill

“Modern-Botok: Custom Dictionary for Modern Tibetan (v0.1),” *Zenodo*, 2024. [doi:10.5281/zenodo.14034747](https://doi.org/10.5281/zenodo.14034747).

Erhard, Franz Xaver, and Xiaoying 笑影

“Foreign Names and Places in Tibetan Newspapers of the 1950s and 1960s,” *Revue d’Etudes Tibétaines* (74), 2025, pp. 172–186.

Erhard, Franz Xaver, Xiaoying 笑影, Robert Barnett, and Nathan W. Hill

“Toponyms and Anthroponyms from Tibetan-language Newspapers of the 1950s and 1960s: Three Name Lists (v1.0),” [Data set]. *Zenodo*, 2024. [doi:10.5281/zenodo.14289491](https://doi.org/10.5281/zenodo.14289491)

Dakpa, Jamyang, Tashi Dhondup, Yeshe Jigme Gangne, Edward Garrett, Marieke Meelen, and Sonam Wangyal.

“Modern Tibetan corpus annotated for verb-argument dependency relations (v1.0),” [Data set]. *Zenodo*, (2021). [doi:10.5281/zenodo.4727129](https://doi.org/10.5281/zenodo.4727129).

Faggionato, Christian, and Edward Garrett.

“Constraint grammars for Tibetan language processing,” *NoDaLiDa 2019 Workshop on Constraint Grammar-Methods, Tools and Applications, Linköping Electronic Conference Proceedings* 168 (3), 2019. Available online at <http://www.ep.liu.se/ecp/168/003/ecp19168003.pdf> (accessed January 24, 2025).

Faggionato, Christian, Edward Garrett, Nathan W. Hill, Samyo Rode, Nikolai Solmsdorf, and Sonam Wangyal.

“Classical Tibetan Corpus Annotated for Verb-Argument Dependency Relations (v1.0),” *Zenodo*, 2021. [doi:10.5281/zenodo.4727108](https://doi.org/10.5281/zenodo.4727108).

Hackett, Paul G.

“Automatic Segmentation and Part-Of-Speech Tagging For Tibetan: A First Step Towards Machine Translation.” In *Proceedings of the 9th Seminar of the International Association for Tibetan Studies*, 2000, pp. 1-18. Available online at <http://hdl.handle.net/10022/AC:P:10471> (accessed on December 18, 2024).

Garrett, Edward, and Nathan W. Hill.

“Constituent order in the Tibetan noun phrase,” *SOAS Working Papers in Linguistics* 17, 2015, pp. 35–48.

Huang, Haoyang, Tianyi Tang, Dongdong Zhang, Wayne Xin Zhao, Ting Song, Yan Xia, and Furu Wei

“Not all languages are created equal in LLMs: Improving multilingual capability by cross-lingual-thought prompting,” *arXiv preprint*, 2023. [doi:10.48550/arXiv.2305.07004](https://doi.org/10.48550/arXiv.2305.07004).

Kyogoku, Yuki, Franz Xaver Erhard, Robert Barnett, and Nathan W. Hill

“TibNorm - Normaliser for Tibetan (Version v1),” *Zenodo*, 2024a. [doi:10.5281/zenodo.10815272](https://doi.org/10.5281/zenodo.10815272)

“Diverge-Gemini POS-tagged Corpus of Modern Tibetan (1.0),” [Data set]. *Zenodo*, 2024b. [doi:10.5281/zenodo.14447192](https://doi.org/10.5281/zenodo.14447192).

“Basic Modern Tibetan SpaCy Model,” *Zenodo*, 2024c. [doi:10.5281/zenodo.10806456](https://doi.org/10.5281/zenodo.10806456).

Li, Yan, Xiaomin Li, Yiru Wang, Hui Lü, Fenfang Li, and La Duo

“Character-based Joint Word Segmentation and Part-of-Speech Tagging for Tibetan Based on Deep Learning,” *Transactions on Asian and Low-Resource Language Information Processing*, 21 (5), Article 95, 2022, pp. 1-15, [doi:10.1145/3511600](https://doi.org/10.1145/3511600).

Lv, Hui, Chi Pu, La Duo, Yan Li, Qingguo Zhou, and Jun Shen

“T-LLaMA: a Tibetan large language model based on

LLaMA2." *Complex and Intelligent Systems* 11 (1), 2025. [doi:10.1007/s40747-024-01641-7](https://doi.org/10.1007/s40747-024-01641-7).

Meelen, Marieke, and Nathan W. Hill

"Segmenting and POS tagging Classical Tibetan using a memory-based tagger," *Himalayan Linguistics*, 16 (2), 2017.

Meelen, Marieke, Élie Roux, and Nathan W. Hill

"Optimisation of the Largest Annotated Tibetan Corpus Combining Rule-Based, Memory-Based, and Deep-Learning Methods," *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)* 20 (1), 2021, pp. 1-11. [doi:10.1145/3409488](https://doi.org/10.1145/3409488).

Niekler, Andreas, Gregor Wiedemann, and Gerhard Heyer

"Leipzig Corpus Miner: A Text Mining Infrastructure for Qualitative Data Analysis." In *Terminology and Knowledge Engineering*, 2014. Available online: <https://hal.archives-ouvertes.fr/hal-01005878> (accessed on December 18, 2024).

Niekler, Andreas; Christian Kahmann, Manuel Burghardt, and Gerhard Heyer

"The interactive Leipzig Corpus Miner. An extensible and adaptable text analysis tool for content analysis." In: *Publizistik* 68 (2-3), 2023, pp. 325–354. [doi:10.1007/s11616-023-00809-4](https://doi.org/10.1007/s11616-023-00809-4).

Sivarajkumar, Sonish, Mark Kelley, Alyssa Samolyk-Mazzanti, Shyam Visweswaran, and Yanshan Wang.

"An empirical evaluation of prompting strategies for large language models in zero-shot clinical natural language processing: algorithm development and validation study," *JMIR Medical Informatics* 12, 2024.

van der Sloot, Ko, and Maarten van Gompel

"LanguageMachines/mbt (v3.11)," *Zenodo*, 2024. [doi:10.5281/zenodo.14500870](https://doi.org/10.5281/zenodo.14500870)

Xiangxiu, Cairang, Nuo Qun, Nuobu Renqing, Trashi Nyima and Qijun Zhao.

“Research on Tibetan Part-of-Speech Tagging Based on Transformer,” *3rd International Conference on Pattern Recognition and Machine Learning (PRML)*, Chengdu, 2022, pp. 315-320. [doi: 10.1109/PRML56267.2022.9882234](https://doi.org/10.1109/PRML56267.2022.9882234).

Appendices

Appendix A: Comparing Expert vs Botok POS-Tagging

	Expert		Botok		Botok passes	Botok errors
0	བཟློན་དོན་	NOUN	བཟློན་དོན་	OTHER	x	
1	གཙོ་བོ་	ADJ	གཙོ་བོ་	ADJ		
2	འི་	ADP	འི་	PART		
3	སློབ་གསོ་	NOUN	སློབ་གསོ་	OTHER	x	
4	ལག་	DET	ལག་	NOUN		
5	གཉེས་པ་	ADJ	གཉེས་པ་	NO_POS	x	
6	ཕྱི་ལོ་འཛུགས་	NOUN	ཕྱི་ལོ་འཛུགས་	NO_POS	x	
7	ཞིབ་ཚགས་	NOUN	ཞིབ་ཚགས་	NO_POS	x	
8	བྱས་	VERB	བྱས་	VERB		
9	ནས་	SCONJ	ནས་	PART		
10	ལག་བསྟར་	NOUN	ལག་བསྟར་	OTHER	x	
11	བྱ་རྒྱུ་	VERB	བྱ་	VERB		
12			རྒྱུ་	NOUN		x
13	ནི་	PART	ནི་	NO_POS	x	
14	མིག་སྣ་	NOUN	མིག་	NOUN		
15			སྣ་	ADV		x
16	འི་	ADP	འི་	PART		
17	བོད་ལྗོངས་	PROPN	བོད་ལྗོངས་	PROPN		
18	ཡོངས་	DET	ཡོངས་	DET		
19	ཀྱི་	ADP	ཀྱི་	PART		
20	ས་གནས་	NOUN	ས་གནས་	OTHER	x	
21	ལག་	NOUN	ལག་	NOUN		
22	དང་	CCONJ	དང་	NO_POS	x	
23	ཚན་པ་	NOUN	ཚན་པ་	OTHER	x	
24	ལག་	NOUN	ལག་	NOUN		
25	གི་	ADP	གི་	PART		

	Expert		Botok		Botok passes	Botok errors
26	གལ་ཚེ	NOUN	གལ་ཚེ	OTHER	x	
27	འི་	ADP	འི་	PART		
28	ལས་དོན་	NOUN	ལས་དོན་	OTHER	x	
29	ཞིག་	DET	ཞིག་	PART		
30	ཡིན	AUX	ཡིན	NO_POS	x	
31		PUNCT		PUNCT		
					13	2

Appendix B Evaluation of Gemini (tokenisation and) POS-tagging

text = ཚོགས་འདུ་འི་ ཐོག་ སྤྱི་ཚབ་ ཟུ་ཅི་ ཞེ་ ཅིན་ མིང་ གི་ དམག་དཔུང་ ལྷོ་བས་ ཚེ་ ར་ གཏོང་བ་ འི་ དགོངས་པ་ དང་ |

#	Token	Flag	POS	Flag
1	ཚོགས་འདུ	TRUE	NOUN	TRUE
2	འི་	TRUE	ADP	TRUE
3	ཐོག་	TRUE	ADP	TRUE
4	སྤྱི་ཚབ་ཟུ་ཅི་	FALSE	NOUN	TRUE
5	ཞེ་ཅིན་མིང་	TRUE	PROPN	TRUE
6	གི་	TRUE	ADP	TRUE
7	དམག་དཔུང་	TRUE	NOUN	TRUE
8	ལྷོ་བས་ཚེ་	FALSE	VERB	FALSE
9	གཏོང་བ་	TRUE	AUX	TRUE
10	འི་	TRUE	ADP	TRUE
11	དགོངས་པ་	TRUE	NOUN	TRUE
12	དང་	TRUE	CCONJ	TRUE
13		TRUE	PUNCT	TRUE

Appendix C Evaluation of Tibetan for spaCy 1.1 (tibetan_tib_en_ver1-0.0.1) tokenisation

text = ཚོགས་འདུ་ འི་ ཐོག་ ལྷི་ལྷའ་ ལུ་ ཅི་ ཞི་ ཅིན་ མིང་ ལྷི་ དམག་དཔུང་ ལྷོ་བས་ ཚེ་ ར་ གཏོང་བ་ འི་ དགོངས་པ་ དང་ །

#	Token	Flag
1	ཚོགས་འདུ	TRUE
2	འི་	TRUE
3	ཐོག་	TRUE
4	ལྷི་ལྷའ་	TRUE
5	ལུ་	FALSE
6	ཅི་	FALSE
7	ཞི་	FALSE
8	ཅིན་	FALSE
9	མིང་	FALSE
10	ལྷི་	TRUE
11	དམག་དཔུང་	TRUE
12	ལྷོ་བས་	TRUE
13	ཚེ་	TRUE
14	ར་	TRUE
15	གཏོང་བ་	TRUE
16	འི་	TRUE
17	དགོངས་པ་	TRUE
18	དང་	TRUE
19	།	TRUE

Appendix D Evaluation of Modern Botok tokenisation

text = ཚོགས་འདུ་ འི་ ཐོག་ ལྷི་ལྷའ་ ལུ་ ཅི་ ཞི་ ཅིན་ མིང་ ལྷི་ དམག་དཔུང་ ལྷོ་བས་ ཚེ་ ར་ གཏོང་བ་ འི་ དགོངས་པ་ དང་ །

#	Token	Flag
1	ཚོགས་འདུ	TRUE
2	འི་	TRUE

#	Token	Flag
3	ཐོག་	TRUE
4	སྤྱི་ལུག་	TRUE
5	རྒྱུ་ཅི་	TRUE
6	ཞི་ཅིན་མིང་	TRUE
7	གི་	TRUE
8	དམག་དཔུང་	TRUE
9	སྟོབས་ཆེར་	FALSE
10	གཏོང་བ	TRUE
11	འི་	TRUE
12	དགོངས་པ་	TRUE
13	དང་	TRUE
14		TRUE

Appendix E Evaluation of Modern Tibetan spaCy (xx_bo_tagger-0.1.2) tokenisation and POS-tagging

text = ཚོགས་འདུ་འི་ཐོག་སྤྱི་ལུག་རྒྱུ་ཅི་ཞི་ཅིན་མིང་གི་དམག་དཔུང་སྟོབས་ཆེར་གཏོང་བའི་དགོངས་པ་དང་།

#	Token	Flag	POS	Flag
1	ཚོགས་འདུ	TRUE	NOUN	TRUE
2	འི་	TRUE	ADP	TRUE
3	ཐོག་	TRUE	NOUN	TRUE
4	སྤྱི་ལུག་རྒྱུ་ཅི་	FALSE	NOUN	TRUE
5	ཞི་ཅིན་མིང་	TRUE	NOUN	FALSE
6	གི་	TRUE	ADP	TRUE
7	དམག་དཔུང་	TRUE	NOUN	TRUE
8	སྟོབས་ཆེར་	FALSE	NOUN	FALSE
9	གཏོང་བ	TRUE	VERB	TRUE
10	འི་	TRUE	ADP	TRUE
11	དགོངས་པ་	TRUE	NOUN	TRUE
12	དང་	TRUE	CCONJ	TRUE
13		TRUE	PUNCT	TRUE

Appendix F Comparing POS tagging by three LLMs (September, 2024)

Text = བརྗོད་པོ་གཞི་མེད་སྐོབ་གཞི་ལག་གཉིས་པ་སྤྲིག་འཛུགས་ཞིབ་ཚུགས་བྱས་ནས་ལག་ལསྟར་བྱ་རྒྱུ་ནི་སྲིག་སྲིད་བོད་ལྗོངས་ཡོངས་ཀྱིས་གནས་ལག་དང་ཚན་པ་ལག་གི་གལ་ཆེད་ལས་རྟོག་ཞིག་ཡིན།

Expert		ChatGPT4				ChatGPT4o				Claude 3.5				Gemini																			
		Token same as expert Y/N	POS-tags	Token same as expert, POS tag is same (Y) or equivalent (Y~) to expert	Token same as expert Y/N	POS-tags	Token same as expert, POS tag is same (Y) or equivalent (Y~) to expert	Token same as expert Y/N	POS-tags	same as expert, POS tag is same (Y) or equivalent (Y~) to expert	Token same as expert Y/N	POS-tags	same as expert, POS tag is same (Y) or equivalent (Y~) to expert	Token same as expert Y/N	POS-tags																		
བཟོད་པོ་	NOUN	N	NOUN		N	VERB		Y	NOUN	Y	Y	NOUN	Y																				
		N	NOUN		N	NOUN																											
གཞི་མེད་	ADJ	N	ADJECTIVE		N	ADJECTIVE		N	ADJECTIVE		N	ADJECTIVE																					
		N	ADP	GEN. PART	N	GEN. PART																											
སྐོབ་གཞི་	NOUN	N	NOUN		N	NOUN		Y	NOUN	Y	Y	NOUN	Y																				
		N	NOUN		N	NOUN																											
ལག་	DET	Y	NOUN	N	Y	NOUN	N	Y	PLURAL	Y~	Y	NOUN	N																				
		Y	ORD. NUMBER	Y~	Y	ADJECTIVE	Y	Y	NUMERAL	Y~	Y	NUMERAL	Y~																				
སྤྲིག་འཛུགས་	NOUN	N	VERB		N	VERB		Y	NOUN	Y	Y	VERB	N																				
		N	VERB		N	VERB																											
ཞིབ་ཚུགས་	NOUN	N	ADJECTIVE		N	ADJECTIVE		Y	ADVERB	Y~	Y	VERB	N																				
		N	NOUN		N	NOUN																											
བྱས་	VERB	Y	VERB	Y	Y	VERB	Y	N	VERB		N	CONJUNCTION																					
		Y	ABL. PART	Y~	Y	ABL. PART	Y~																										
ལག་ལསྟར་	NOUN	N	NOUN		N	NOUN		Y	NOUN	Y	N	NOUN																					
		N	VERB		N	VERB																											
བྱ་རྒྱུ་	VERB	N	NOUN		N	VERB		Y	VERB NOMINAL	Y~																							
		N	NOUN		N	NOUN																											
ནི་	PART	Y	VERB. AUX PART	N	Y	NOM. PART	Y~	Y	TOPIC MARK.	Y~	Y	COPULA	N																				
		N	NOUN		N	NOUN		N	ADJECTIVE		N	ADJECTIVE																					
སྲིག་པུ་	NOUN	N	NOUN		N	NOUN		N	ADJECTIVE		N	ADJECTIVE																					
		N	ADJECTIVE		N	NOUN																											
དེ་	ADP	Y	GEN. PART	Y~	Y	GEN. PART	Y~																										
		N	NOUN		N	NOUN		Y	NOUN	Y~	Y	PROPER NOUN	Y																				
བོད་ལྗོངས་	PROPN	N	NOUN		N	NOUN																											
		N	NOUN		N	NOUN																											
ཡོངས་	DET	Y	NOUN	N	Y	ADJECTIVE	N	Y	ADVERB	N	N	ADJECTIVE																					
		Y	GEN. PART	Y	Y	GEN. PART	Y	Y	GEN. PART	Y																							
ཀྱི་	ADP	N	NOUN		N	NOUN		Y	NOUN	Y	Y	NOUN	Y																				
		N	NOUN		N	NOUN																											
ལག་	DET	Y	NOUN	N	Y	NOUN	N	Y	PLUR. MARK.	Y~	Y	NOUN	N																				
		Y	CONJ.	Y	Y	CONJ.	Y	Y	CONJ.	Y~	Y	CONJ.	Y																				
དང་	CCONJ	Y	CONJ.	Y	Y	CONJ.	Y	Y	CONJ.	Y~	Y	CONJ.	Y																				
		Y	NOUN	Y	Y	NOUN	Y	Y	NOUN	Y~	Y	NOUN	Y																				
ཚན་པ་	NOUN	Y	NOUN	Y	Y	NOUN	Y	Y	NOUN	Y~	Y	NOUN	Y																				
		Y	DET	Y	Y	NOUN	N	Y	PLUR. MARK.	Y~	Y	NOUN	N																				
ལག་	DET	Y	NOUN	N	Y	NOUN	N	Y	PLUR. MARK.	Y~	Y	NOUN	N																				
		Y	ADP	GEN. PART	Y	Y	GEN. PART	Y	Y	GEN. PART	Y~	Y	GEN. PART	Y~																			
གལ་ཆེ་	NOUN	N	NOUN		N	NOUN		N	ADJECTIVE		N	ADJECTIVE																					
		N	ADP	ADJECTIVE		Y	GEN. PART	Y~																									
དེ་	ADP	N	ADJECTIVE		N	ADJECTIVE																											
		N	NOUN		N	NOUN		Y	NOUN	Y	Y	NOUN	Y																				
ཡམས་པོ་	NOUN	N	NOUN		N	NOUN																											
		N	NOUN		N	NOUN																											
སྲིག་	DET	Y	DEFIN. ART.	Y~	Y	NOMINAL	N	Y	INDEF. ART.	Y~	Y	DETERM.	Y																				
		Y	AUX	Y	Y	VERB	Y~	Y	VERB	Y~	Y	COPULA	Y~																				
		Tags	Tags	Tags	Tags	Tags	Tags	Tags	Tags	Tags	Tags	Tags	Tags	Tags	Tags	Tags																	
		Y=15	Y~10	Y=16	Y~11	Y=11	Y=21	Y=20	Y=17	Y=11	Y=11	Y=11	Y=11	Y=11	Y=11	Y=11																	
		N=24	N=5	N=24	N=5	N=4	N=4	N=1	N=6	N=6	N=6	N=6	N=6	N=6	N=6	N=6																	
Error score relative to expert		ChatGPT4				ChatGPT4o				Claude 3.5				Gemini																			
Tokens		N=24				N=24				N=4				N=6																			
Tags		N=5				N=5				N=1				N=6																			
Errors as %		61,5%				33,3%				60,0%				31,3%				16,0%				4,8%				26,1%				35,3%			

It must be noted that the above comparison, although providing a good overview of the tokenisation and POS tagging capabilities of the tested LLMs, is a snapshot of September 2024. The situation now, at the time of writing a few months later, has changed significantly. Most models have shown improved capabilities in translating and dealing with Tibetan. The situation was also different in June 2024 – when most of the training data for the Divergent Discourses project was created. Then Gemini Pro 1.5 performed best on Tibetan POS tagging tasks. While we had our script creating annotated training data with Gemini Pro 1.5 using Google's cloud API, Anthropic released its newest Claude 3.5 Sonnet model, a powerful alternative for Tibetan language tasks, on June 20, 2024, and an upgrade on October 22, 2024. Although we could not use Claude 3.5 to create training data, we wanted to include it in this comparison to show its good performance in Tibetan language-related tasks.

