A Universal Dependency Treebank for Classical Tibetan*

Christian Faggionato
(University of Cambridge)

The Universal Dependencies (UD) Project's goal is to create a set of multilingual standardised dependency treebanks that are built according to a universal annotation scheme. The present paper describes the work behind the creation of the first Classical Tibetan UD treebank that involves a a semi-automated NLP pipeline, with the implementation of a rule-based dependency parser written in the Constraint Grammar (CG-3) formalism.

1. *Introduction to Dependency Grammar*

Dependency grammar is a type of grammatical framework that focuses on the relationships between words in a sentence. It provides an alternative to phrase structure grammar and generative grammar, which both define a sentence as a set of constituents or phrases. Instead, dependency grammar defines the relationships between words in terms of dependency, where one word (the head) is the main word, and the other word(s) (the dependents) provide additional information about the head. Dependency grammar can be represented using a tree structure, where the root of the tree represents the main verb in the sentence, and the other words in the sentence are attached as dependents to the root or to other words in the sentence (Matthews et al. 1981).

One of the main advantages of dependency grammar is its simplicity and flexibility. In a dependency grammar, each word in a sentence is treated as a separate unit and is assigned a grammatical function, such as subject, object, or modifier. These functions are indicated by the dependencies between the words, rather than by the placement of the words within a phrase or clause.

* Christian Faggionato, "A Universal Dependency Treebank for Classical Tibetan", *Revue d'Etudes Tibétaines*, no. 72, Juillet 2024, pp. 52-69.

This work was funded by the Arts and Humanities Research Council (AHRC), UKRI, as part of the project "The Emergence of Egophoricity: a diachronic investigation into the marking of the conscious self." Project Reference: AH/V011235/1. Principal Investigator: Nathan Hill, SOAS University of London.

This means that the structure of a sentence can be represented as a set of directed dependencies between the words, rather than having a hierarchical structure (Jurafsky & Martin 2014). Figure 1 shows an example of a Classical Tibetan sentence with dependency parsing:

(1) ཨ་ཁུ་ དང་ ཨ་ནེ་ ལ་ ཤ་ཁོག་ འཛུངས།

a-khu	*dang*	*a-ne*	*la*	*sha-khog*
uncle.NOUN	and.ADP	aunt.NOUN	to.ADP	carcass.ADP
'dzungs	/			
gave.VERB				

'We presented an [entire animal] carcass to my aunt and uncle (de Jong (1959), p. 33 ln. 14)'

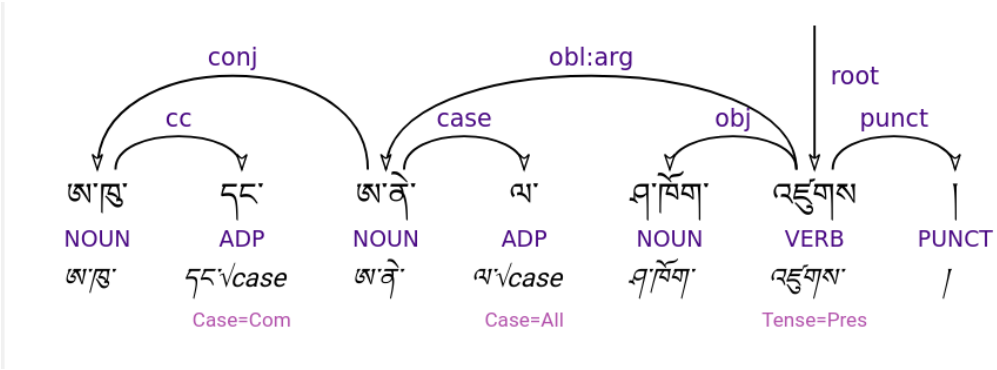

Fig. 1

In this example relations among the words are illustrated above the sentence by labelled arcs from heads to dependents; labels are drawn from a fixed set of grammatical relations and the head of the entire sentence structure is marked by a root node. Each token has three layers of information: word, Part-of-Speech tag (POS) and lemma.

2. *The Universal Dependencies Project*

The Universal Dependencies (UD) treebank project is a collaborative effort that started in 2014 and it is aimed at creating a consistent representation of syntactic dependencies across multiple languages. This is achieved by defin-

ing a set of universal dependencies (Nivre, de Marneffe, Ginter, Hajič, Manning, Pyysalo, Schuster, Tyers & Zeman 2020).[1]

The UD treebank project has been successful in creating annotated corpora for many languages including low-resourced ones. Each treebank is annotated using a common annotation scheme, which includes information about the words in the sentence, their part-of-speech tags, and the relationships between them in order to capture the underlying grammatical structure. Additionally, all the annotated corpora are available for free and can be easily accessed through the UD treebank website. This common annotation scheme generally allows for cross-lingual comparison and analysis, making it easier to develop NLP models that can be used across multiple languages and for a variety of NLP tasks, such as machine translation, text classification, and named entity recognition. Furthermore dependency relations can also provide information on the semantic relationship between predicates and their arguments which is useful for other NLP applications such as question answering and information extraction.[2]

The first dependency parsed corpus of Classical Tibetan –using the guidelines of the UD treebank project– was compiled during two AHRC funded projects that took place at the School of Asian and African Studies (SOAS) in London. During the first project, "Tibetan in Digital Communication" (2012-2015), a group of four Classical Tibetan texts were manually POS tagged: the *mdzang blun* མཛངས་བླུན་ཞེས་བྱ་བའི་མདོ་, "The Sutra of the Wise and the Foolish", the *Mi la'i rnam thar* མི་ལའི་རྣམ་ཐར་, "The Biography of Milarepa", the *Mar pa lo tsā' i rnam thar* མ་རཔ་ལོ་ཙཱ་ཱི་རྣམ་ཐར་, "The Biography of Marpa", and the *Bu ston chos ' byung* བུ་སྟོན་ཆོས་འབྱུང་, "The History of Buddhism" by *Bu ston Rin chen grub*. During the follow up project "Lexicography in Motion" (2017-2021), the corpus was expanded with two Old Tibetan texts, the *Old Tibetan Annals* and the *Old Tibetan Chronicle* and an additional Classical Tibetan text, the *Twa ra nwa tha' i rgya gar chos ' byung* ཏྭ་ར་ནྭ་ཐའི་རྒྱ་གར་ཆོས་འབྱུང་, "the History of Buddhism" by *Tāranātha Kun dga' Snying po*. All the new texts have been automatically POS tagged using the method developed by Meelen, Roux & Hill (2021) and then manually corrected.[3] During the same project we compiled a diachronic lexicon of Tibetan verbs and the dependency relations linking verbs to their arguments were manually annotated using the following scheme:

1 https://universaldependencies.org/

2 Nivre, de Marneffe, Ginter, Goldberg, Hajič, Manning, McDonald, Petrov, Pyysalo, Silveira, Tsarfaty & Zeman (2016).

3 A full manual correction of the POS tagging was only done for the two Old Tibetan texts: at the present stage, this small Old Tibetan corpus represents the Gold Standard, which can be used for training.

- *arg1* (changed to *nsubj* in accordance with UD annotation scheme): the first argument or " subject" of a verb, which may be agentive or unmarked, but not oblique.
- *arg2* (changed to *obj* in accordance to the UD annotation scheme): the second argument or " object" of a verb, which cannot be oblique.
- *arg2-lvc* (changed to extitobj-lvc in accordance to the UD annotation scheme): the second argument of a verb, which together with it forms a complex predicate.
- *argcl*: the clausal argument of a verb.
- *obl-adv*: an oblique marked nominal, which behaves like an adverb.
- *obl-arg*: an oblique marked nominal, which is considered an argument of the verb.
- *obl*: an oblique nominal that modifies a verb.

The process of manually annotating verbal arguments gave rise to the idea of developing an NLP tool that can automatically map missing dependencies for other sentence constituents.

3. *A Dependency Constraint Grammar for Classical Tibetan*

The constraint grammar formalism (CG) is a rule-based formalism for writing disambiguation and syntactic annotation grammars, originally introduced by Karlsson (Karlsson, Voutilainen, Heikkila & Anttila 1995) and successively implemented with a set of rules that creates dependency annotation (CG-3). Its VISL constraint grammar compiler (version 3) (VISL-CG3) implemented in the IDE for CG-3, is used for the compilation of constraint grammar rules. The constraint grammar analyzes the texts with a bottom-up scanning. Every disambiguation is solved step by step with the help of morpho-syntactic context. Constraint-grammar rules usually contain context conditions, domains, operators and targets. The context can be absolute, referring to a fixed token position within the text, or relative, referring to a token to the left or right with a certain distance to a specific constraint. We can modify the context using barriers made of tokens or SET of tokens that stop the scanning of the sentence. Furthermore, we can link context to other context with the LINK rule. In this way the constraint grammar works globally and creates complex syntactic relations.

Dependency treebanks can be created using human annotators, or using automatic parsers to provide an initial parse and then having annotators to

correct the parses. To facilitate the creation of the first dependency parsed corpus of Classical Tibetan I decided to write a a rule-based parser, in the CG-3 constraint grammar formalism, able to generate full dependencies from the verb arguments dependencies we already had at our disposal. I created 72 SETPARENT rules and 41 mapping rules that generate case-marking relations, dependencies for noun phrases—linking modifiers such as adjectives, determiners and demonstratives to head nouns—and dependencies linking converbs, punctuation and adverbs verbs.

In the CG-3 formalism the dependency analysis is done using specific rules, i.e. SETPARENT (mapping a token to its parent) and SETCHILD (mapping a token to its child). There are also rules used to correct and fix errors, i.e. ADDCOHORT (adding a token and all its readings), MOVECOHORT (moving a token and all its readings) and DELETE (deleting a token with all its readings). The grammar at the moment uses a set of POS tags and and the dependency tags for verbal arguments to generate a full set of dependency relations tags for all the sentence constituents. In order to establish dependency relations, the dependency tags are expressed in the following way, 5->2: the first digit indicates the absolute position of the token in the sentence and it points to the absolute position of the token representing the mother. Usually a verb points to 0, which indicates its head status (Bick & Didriksen 2015).

I tested the CG-3 dependency grammar on both Old Tibetan and Classical Tibetan texts. I opted to work also on the Old Tibetan texts due to practical considerations, as these texts represented our gold standard in terms of POS tagging and annotation of verb arguments. It is worth noting that Old Tibetan and Classical Tibetan are very similar in terms of grammar and vocabulary, but they differ sustantially in terms of spelling and orthography.Dotson & Helman-Ważny (2016) I developed a python script in order to normalize Old Tibetan to Classical Tibetan, to make the two languages to look similar.[4] An improved version of the normalization grammar is now implemented in the pre-processing python script of the NLP pipeline developed by Faggionato, Hill & Meelen (2022). The normalization process allowed the analysis of Old Tibetan with the NLP tools for segmentation and POS tagging available for Classical Tibetan (Faggionato & Meelen 2019).

The Tibetan texts have been first exported in the CoNNL-U format, where each line has 10 fields, separated by tabs, containing information for every word/token such as word index or ID, word form, lemma, universal POS tag, a list of morphological features, the head of the current word (which is either a ID value or 0) the universal dependency relation to the head and other annotations. Figure 2 shows an example of a sentence in the CoNNL-U

4 https://github.com/lothelanor/actib/blob/main/preprocessing.py

format. After each sentence there is always an empty line which represents sentence boundaries.

```
<s ref="T112">
1   རྒྱུ་རྐྱེ་  རྒྱུ་རྐྱེ་  NOUN    _   Number=Sing 0   root    _   _
2   ལ་  ཀྱི་√case  ADP _   Case=Gen    0   root    _   _
3   སྐྱེ་  སྐྱེ་  NOUN    _   Number=Sing 0   root    _   _
4   ངོ་  ཀྱི་√case  ADP _   Case=Gen    0   root    _   _
5   དལ་བུབ་  དལ་བུབ་  NOUN    _   Number=Sing 7   arg2    _   _
6   ཁྱད་  ཁྱད་  NOUN    _   Number=Sing 7   obl-adv _   _
7   བསད་ གསོད་VERB    _   Tense=Past  0   root    _   _
8   ནས  ནས་√cv  SCONJ    _   Case=Ela    0   root    _   _
9   |   |   PUNCT   _   _   0   root    _   _
10  |   |   PUNCT   _   _   0   root    _   _
</s>
```

Fig. 2 – CoNLL-U Format

The CG-3 input files have been created modifying the existing CoNNL-U files and retaining information such as ID, POS, lemma, dependencies for verb arguments, dependency tags and other syntactic features. Example (2) shows an extract from a VISL-CG3 input file with five tokens:

(2) "<ནམ་མཁའ་>"
"ནམ་མཁའ་" NOUN Number=Sing @obl-arg #1->3
"<ལས་>"
"ལས་√case" ADP Case=Abl @root #2->0
"<བཙས་>"
"འབད་√1" VERB Tense=Past @root #3->0
"<ཏེ>"
"ཏེ་√cv" SCONJ Case=Sem @root #4->0
"<|>"
"|" PUNCT @root #5->0

Example (3) shows the output file containing the dependencies and dependency labels generated by the CG-3 grammar. The generated dependencies link all the tokens within a sentence to the root element which is usually a verb or a head noun:

(3) "<ནམ་མཁའ་>"
"ནམ་མཁའ་" NOUN Number=Sing @obl-arg #1->3
"<ལས་>"
"ལས་√case" ADP Case=Abl @case #2->1
"<བབས་>"
"འབབ་√1" VERB Tense=Past @root #3->0
"<ཏེ>"
"ཏེ་√cv" SCONJ Case=Sem @mark #4->3
"<|>"
"|" PUNCT @punct #5->3

The CG-3 output file is then converted again into the CoNLL-U format and uploaded into Arboratorgrew.[5] Arboratorgrew is a popular software tool for linguistic analysis that provides a graphical representation of the tree structure of sentences, and makes it very easy and quick to manually correct dependency relations (Guibon, Courtin, Gerdes & Guillaume 2020).

The CG-3 dependency grammar is made of three main sections, and each of them serves a distinct purpose in generating the dependency treebank.

In the first section I defined the sentence boundaries creating a sentence break after specific converbs and cases: final སོ་, semifinal ཏེ་, imperative ཤིག་, imperfective ཞིང་, na-re ན་རེ་, question འམ་ and associative དང་.[6] Furthermore I introduced sentence boundaries when verbs are not followed by any converb but are followed by punctuation markers such as *shads*. The punctuation condition is necessary because we do not want to introduce any sentence segmentation when we have a chain of two or more verbs. In the same section I introduced all the POS tags and the universal dependency tags used by the grammar.

In the second section I defined three sets of helpers. Each set creates boundaries for the noun phrases, define their constituents and their nominal heads. Example (4) shows the three sets in the CG-3 formalism:

(4) SET np.elem = (NOUN) OR (ADJ) OR (NUM) OR (PROPN) OR (DET) OR (PRON) OR VN;
SET LEFT_NP_BOUNDARY = (VERB) OR (ADP) OR (SCONJ) OR (PART) OR (ADV) OR (AUX) OR (PUNCT);
SET RIGHT_NP_BOUNDARY = (ADP) OR (SCONJ) OR (Polarity=Neg) OR (PUNCT);
SET Head_NOUN = (NOUN) OR (NUM) OR (PROPN) OR (PRON) OR (DET) OR VN;

5 https://arboratorgrew.elizia.net/#/
6 Associative cases are sentence boundaries only if they follow verbal nouns.

In the third and main section, I created the dependency relations for all the tokens, using the CG-3 SETPARENT and MAP rules. A first set of rules deals with punctuations, linking them to the root verb, usually on the left of the PUNCT token. In this section I further improved the CG-3 dependencies linking subordinate clauses to the main verbs with the tag *advcl* and using the new sentence segmentation rules and annotation scheme we recently developed (Faggionato, Meelen & Hill 2023). For a detailed description of the newly developed NLP pipeline for processing Old and Classical Tibetan texts see Faggionato et al. (2022). Figure 3 shows an example of a subordinate clause linked to the main verb with the correct dependency relation *advcl*.

(5) བཞེངས་ནས་ཕྱི་ར་རྒྱངས་བཀྱེད་དེ་

bzhengs	*nas*	*phy*	*r*	*rgyangs*
get_up.VERB	SCONJ	outside.DET	ADP	far.NOUN
bkyed	*de*			
move_away.VERB	SCONJ			

'[the lady] got up in order to move far away' (de Jong (1959), p. 78 ln. 25-26)'

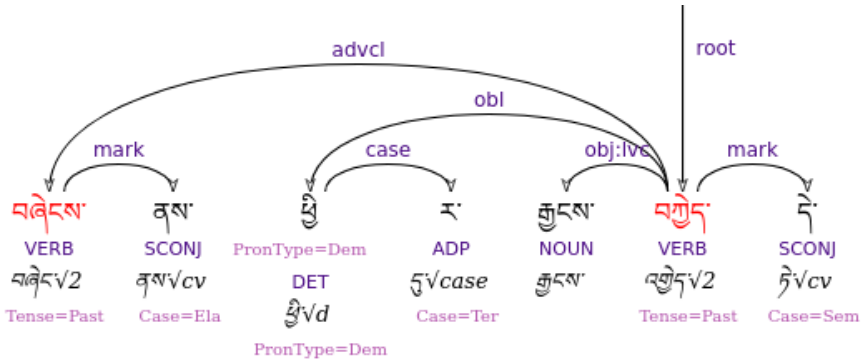

Fig. 3 – Subordinate Clause

In the main section I also created the CG-3 rules that deal with the internal dependencies of the noun phrase (NP) elements. Analyzing the distributions and common patterns of the Tibetan NP constituents, I created a set of dependency rules that solve these potential issues. These rules prevent the parser to create wrong cross-dependencies between sequences of nouns (Garrett & Hill 2015).

Patterns of head nouns followed by one or more determiner, numeral, pronoun or adjective and followed by another head noun, are not a real challenge for the parser. Also cases of head nouns, already tagged as verb arguments, followed by nouns that function as appositions are easily solved by the CG-3 dependency rules. Figure 4 shows a set of dependency relations correctly generated for the internal elements of a NP.

(6) མ་སྨད་ གསུམ་ ཀ་ ས་ དྲྭ་ སོ་

ma-smad *gsum* *ka* *s* *ngus* *so*
mother.NOUN three.NUM all.DET ADP cry.VERB PART
'the three of us, mother and children, cried' (de Jong (1959), p. 37 ln. 6)

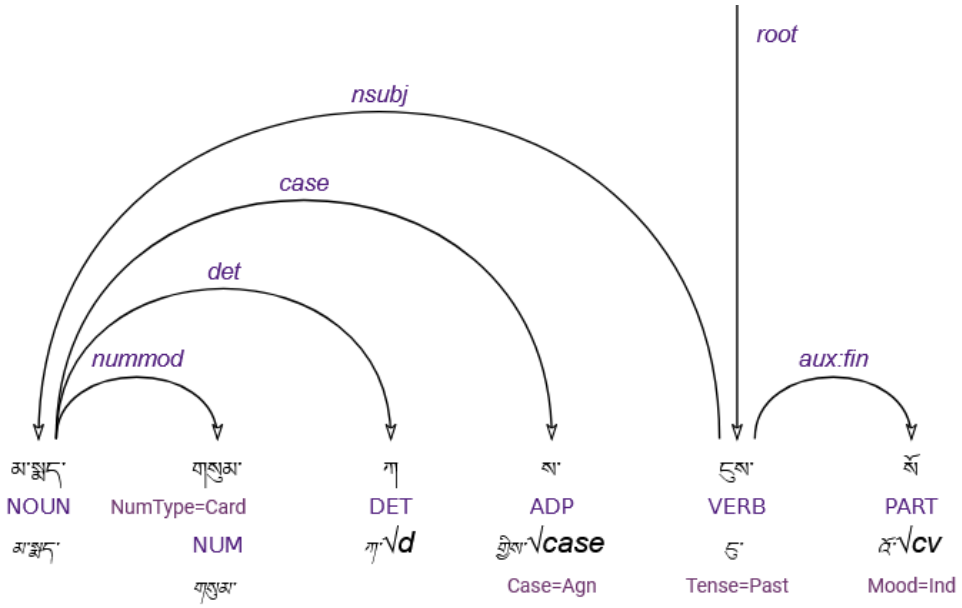

Fig. 4 – NP Elements

The CG-3 rules that creates these dependencies follow the same structure for determiners, numerals, adjectives, proper nouns and nouns that function as appositions. Here is the set of rules that creates the dependencies for nouns in apposition:

(7) SETPARENT (NOUN) (NONE p ALLPOS) TO (-1* Head_NOUN + TAGS OR (ADJ) + TAGS BARRIER LEFT_NP_BOUNDARY);

SETPARENT (NOUN) (NONE p ALLPOS) TO (-1* Head_NOUN BARRIER LEFT_
NP_BOUNDARY)(-1 LEFT_NP_BOUNDARY);

SETPARENT (NOUN) (NONE p ALLPOS) TO (-1* Head_NOUN BARRIER LEFT_
NP_BOUNDARY)(-2< (PUNCT))(NONE p ALLPOS - TAGS);

MAP (@appos) TARGET (NOUN) - TAGS (p Head_NOUN OR (ADJ) + TAGS);

The first SETPARENT rule of the set deals with cases such as NOUN + Head_ NOUN + ADJ in order to create a parental relation between the adjective and the second noun which has been manually tagged as a verb argument. Figure 5 shows the dependencies created by the first SETPARENT rule:

(8) གཡོན་ དུ་ ཐལ་བ་ སྤྱར་ གང་ ཁྱེར །

gyon	*du*	*thal-ba*	*spar*	*gang*
left.NOUN	in.ADP	ashes.NOUN	hand.NOUN	full.ADJ
khyer				
carry.VERB	PUNCT			

'carrying a handful of ashes in her left hand' (de Jong (1959), p. 36 ln. 18-19)

The second and third SETPARENT rule deals with other cases were the head noun is not an argument: the condition (-1 LEFT_NP_BOUN -DARY) forces to create a parental relation between the NP constituents and the leftmost element of the NP. The third SETPARENT rule works exactly as the second SETPARENT rule but deals with NPs at the beginning of a new sentence. Figure 6 shows the dependencies generated by these two SETPARENT rules:

(9) ནས་ ཕུལ་ གང་ ངམ

nas	*phul*	*gang*	*ngam*
barley.NOUN	handful.NOUN	full.ADJ	PART

'A handful of barley?' (de Jong (1959), p. 34 ln. 5)

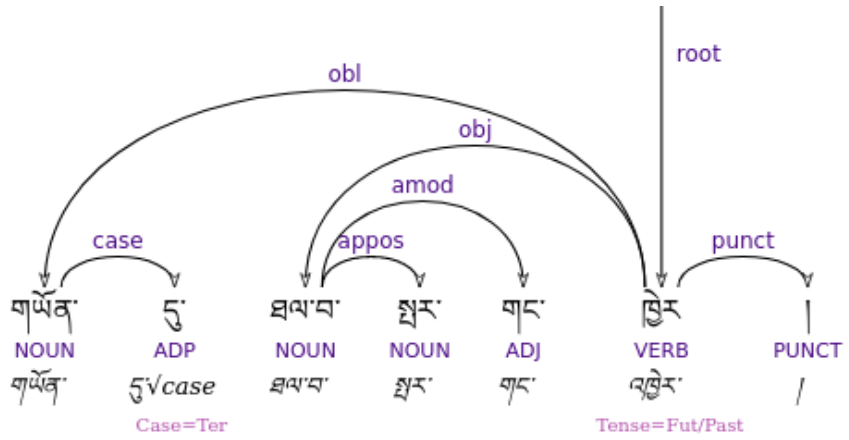

Fig. 5 – Nouns in Apposition - Rule 1

In all the three rules the MAP rule assigns the *adj* tag to the dependencies created with the three SETPARENT rules.

All the cases where nouns are in apposition to each other and are not already manually tagged as verb arguments might require manual correction in the post-processing phase.

In this case a careful analysis of the distribution or verb arguments in absolutive case might help. In fact we would expect zero-marked arguments being positioned as close as possible to the root verb, while other nominals that functions as oblique further away in the sentence. Also, when we have head nouns in absolutive case following each other, as shown in Figure 7, there will be often an intervening clitic, agentive or other case markers. All these considerations, highly reduce the chance of errors for the CG-3 parser.

(10) ཡུམ་ ཡང་ སྤྱན་ཆབ་ འདོན་ ཞིང་

yum	*yang*	*spyan-chab*	*'don*	*zhing*
wife.NOUN	PART	tear.NOUN	expel.VERB	SCONJ

'the wife burst into tears' (de Jong (1959), p. 67 ln. 23)

After the first set of rules for the NP elements, I created a second set of dependency rules for ADP (cases), PART (negations and focus clitics), SCONJ (converbs), and nouns linked to head nouns through the genitive case or the as-

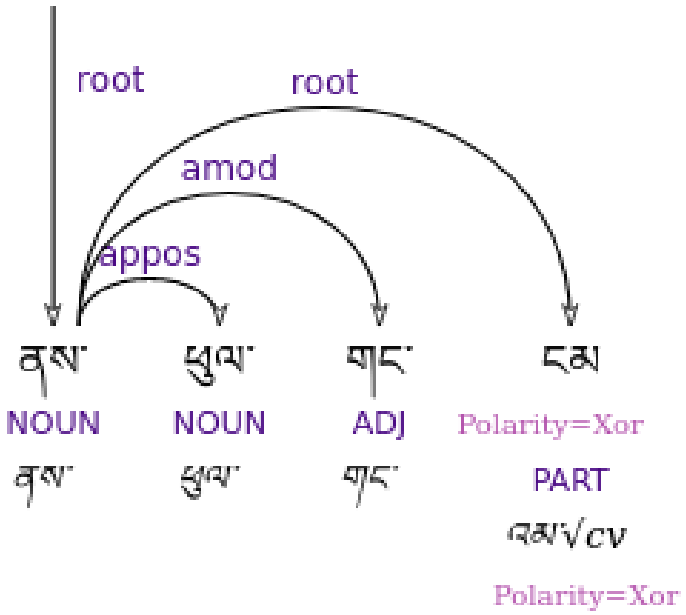

Fig. 6 – Nouns in Apposition - Rule 2 & 3

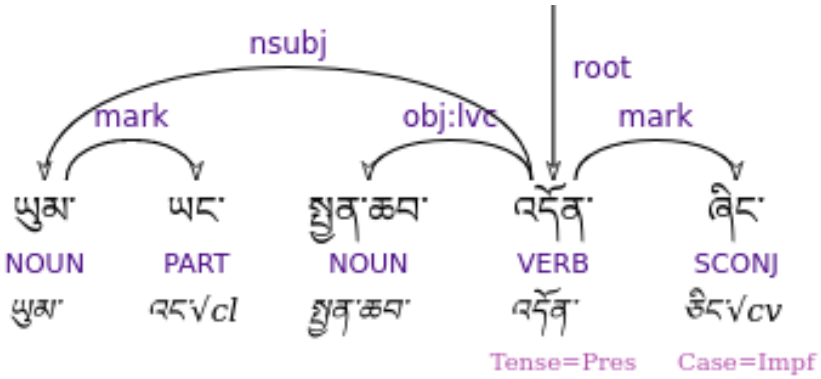

Fig. 7 – Head Nouns in Absolutive Case

sociative case: these nouns are tagged in the dependency treekbank as *nmod*, noun modifiers (see Fig. 8).

Since the parser follows the dependency rules in a sequential order, I was also able to link case markers and focus clitics straight to the head nouns, as

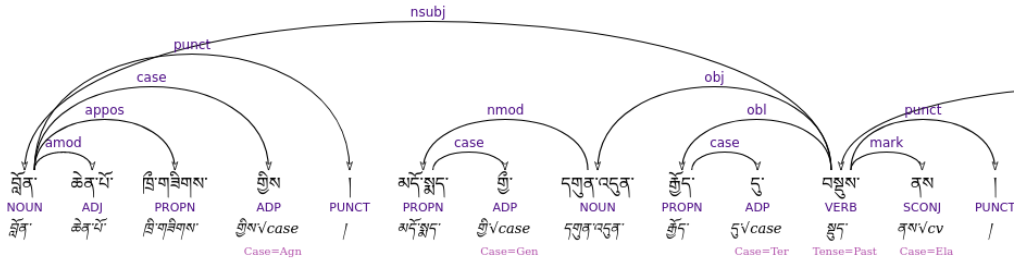

Fig. 8 – Cases and Converbs

all the other elements within the NP already have a dependency relation with their heads. Figure 8 also shows an agentive *gyis* (གྱིས་) correctly linked to its head noun *blon* (བློན་), 'minister'. This has been possible adding a (NONE p ALLPOS - TAGS) condition to the SETPARENT rule: the condition allows the parser to skip all the NP elements that already have a parental dependency relation with any head noun.

(11) བློན་ ཆེན་པོ་ ཁྲི་གཟིགས་ གྱིས ། མདོ་སྨད་ གྱི་ དགུན་འདུན་ རྒྱོད་ དུ་ བསྡུས་ ནས །

blon *chen-po* *khri-gzigs* *gyis* /
minister.NOUN great.ADJ Khri-gzigs.PROPN ADP PUNCT
mndo-smad *gyi* *dgun-'dun* *rgyod*
Mdo-smad.PROPN of.ADP winterNOUN Rgyod.PROPN
du *bsdus* *nas* /
at.ADP convened.VERB after.SCONJ PUNCT
'After Chief minister [Dba' s] Khri-gzigs convened the Mdo-smad winter council at Rgyod' (The Old Tibetan Annals, (Dotson & Hazod 2009))

In the final section of the CG-3 grammar I created a rule for relative clauses, tagged as *acl:rel*, and a section with specific rules targeting particular cases of tokens not mapped by previous dependency rules. Here is the SETPARENT rule that generates the dependencies for relative clauses:

(12) SETPARENT VN - TAGS (NONE p ALLPOS) TO (1* Head_NOUN BARRIER LEFT_NP_BOUNDARY - (Case=Gen) OR TAGS) (-1 (Case=Gen));
MAP (@acl:rel) TARGET VN - TAGS (p Head_NOUN);

The rule links a verbal noun followed by a genitive case to the head noun to its right. Figure 9 shows the dependencies created by this SETPARENT rule.

(13) ཨ་ཁུ་ དང་ ཨ་ནེ་ ས་ གཙོ་ བྱས་པ་ འི་ གཉེན་ཉེ་འཁོར་ །

a-khu *dang* *a-ne* *s* *gtso*
uncle.NOUN and.ADP aunt.NOUN ADP main.NOUN
byas-pa *'i* *gnye-nye-'khor* */*
made.VERB that.ADP close_relatives.NOUN PUNCT
'[our] close relatives, that were headed by my [paternal] aunt and uncle' (de Jong (1959), p. 31 ln. 12)

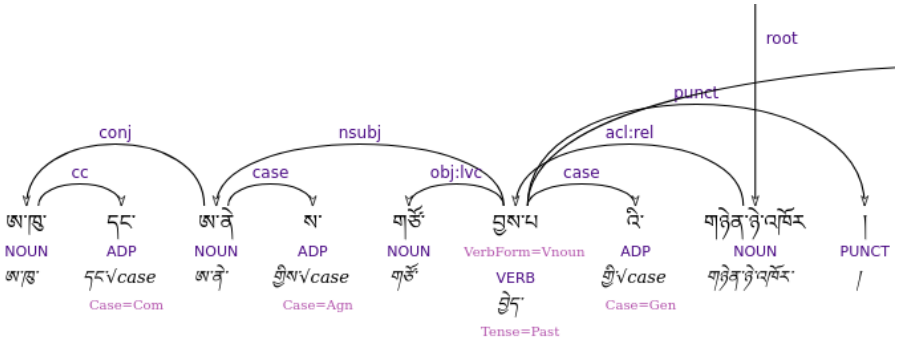

Fig. 9 – Relative Clause

4. *Evaluations and Future Improvements*

To evaluate the results of the dependency parser, I manually corrected the whole text of the Old Tibetan Annals and some sections of the *Mi la'i rnam thar*, "The Biography of Milarepa", for a total of 430 sentences and 12k tokens. The CG-3 dependency grammar achieves an accuracy level of 80% in creating dependencies and assigning dependency labels to unlabelled tokens. Additionally, the parser performs exceptionally in recognising subordinate clauses with an accuracy level of 90%. As it is expected, No automatic NLP task is

100% correct and a certain degree of manual correction is needed. The overall process of generating a full dependency treebank for Classical Tibetan it is hugely facilitated by the CG-3 dependency parser.

After the analysis of the obtained results, I identified some recurring error patterns. As already pointed out in the previous section, the challenge posed by chain of nouns leaves some of the tokens without a dependency tag. Again, this issue could be partially solved adding a layer of animacy information that will help in disambiguating head nouns. It is worth noting that according to the new and improved word segmentation guidelines that we recently developed [7]words have been split as much as possible to create more insight into the internal structure of the sentences, and that creates an additional problem with personal names and titles in terms of generating automated dependencies. Some manual correction will be needed in all these cases.

A second case where the CG-3 dependency parser needs improvement is represented by direct speech sentences that are not ending with a question converb or case. A possible approach is to use the POS tags for quotative clitics together with a lookup rule into a list of verbs of speech and create, in the first section of the CG-3 grammar, sentence boundaries after specific constructions such as *smras pa* སྨྲས་པ་, 'it is said', or *la gsol pa* ལ་གསོལ་པ་, 'said, replied'.

To create the first version of the UD Treebank for Classical Tibetan, I am curating a corpus of almost 300 sentences, carefully selected from three different texts. This approach will ensure that the corpus offers a well-rounded representation of the language from a diachronic point of view. The three texts in chronological order are: The Old Tibetan Annals (9th c.), The Old Tibetan Chronicle (10th c.) and the *Mi la'i rnam thar*, "The Biography of Milarepa" (15th.). After its validation, the treebank will be submitted and deposited in the UD project website for public use under the Creative Commons Attribution-ShareAlike (CC BY-SA) license.[8] All the material including the CG-3 grammars and the annotated CONLL-U files will be available as soon as they are ready at my Github UD_Tibetan repository.

Once the first version of the treebank is completed, it will be possible to train and test neural dependency parsers, like StanfordNLP, a transition-based neural parser trainable with CONLL-X files and word embeddings, and the UDPipe parser, a transition-based parser using a neural-network classifier which provides good results with small UD Treebanks. The UDPipe pipeline

7 The Segmentation and POS manual for Classical Tibetan are available on Zenodo at https://zenodo.org/records/7880130

8 https://universaldependencies.org/

is easily trainable on new languages with training data in CoNLL-U format and does not require additional resources such as morphosyntactic dictionaries or any feature engineering (Straka, Hajič & Straková 2016). At the same time I aim to expand the UD Treebank by incorporating additional texts and sentences, enhancing the accuracy and precision of the dependency parsers and improving their performance on a wider range of sentence structures and linguistic phenomena.

5. Conclusions

This paper outlines the procedures involved in developing a fully-annotated dependency treebank for Classical Tibetan. The process has been partially automated through the implementation of a CG-3 rule-based dependency parser. The data output provides a foundational framework, essential for the training of any neural model aimed at improving automated dependency parsing for the language.

The development of the Classical Tibetan UD corpus represents a significant contribution to both the linguistic and computational communities. For linguists, the corpus allows for a more comprehensive understanding of the grammatical structures of Classical Tibetan. It provides a wealth of data that can be used to analyze and describe the syntax and morphology of the language, including its word order and case marking. This data can be used to test linguistic theories and hypotheses, and to gain deeper insights into the nature of the language and its historical development. From an NLP point of view, the corpus has the potential to significantly improve the accuracy of NLP tools and applications for Classical Tibetan. With a complete dependency treebank, computational models can be trained to accurately parse and analyze Classical Tibetan texts, enabling the development of technologies such as machine translation and language generation. Furthermore, the corpus can be used to develop language models that can assist in automatic speech recognition, sentiment analysis, and other NLP applications.

Overall, the development of a full dependency corpus for Classical and Old Tibetan provides a valuable resource for scholars, researchers, and language enthusiasts interested in understanding and analyzing the language, as well as for developers seeking to build robust NLP tools and applications for Classical Tibetan.

Bibliography

Bick, Eckhard & Tino Didriksen. 2015. CG-3 — beyond classical constraint grammar. In *Proceedings of the 20th nordic conference of computational lin-*

guistics (NODALIDA 2015), 31–39. Vilnius, Lithuania: Linköping University Electronic Press, Sweden. https://aclanthology.org/W15-1807.

Dotson, B. & G. Hazod. 2009. *The Old Tibetan Annals: An Annotated Translation of Tibet's First History* Denkschriften (Österreichische Akademie der Wissenschaften. Philosophisch-Historische Klasse). Verlag der osterreichischen Akademie der Wissenschaften.

Dotson, B. & A. Helman-Ważny. 2016. *Codicology, paleography, and orthography of early tibetan documents: Methods and a case study* Wiener Studien zur Tibetologie und Buddhismuskunde. Arbeitskreis für Tibetische und Buddhistische Studien Universität Wien.

Faggionato, Christian, Nathan Hill & Marieke Meelen. 2022. NLP pipeline for annotating (endangered) Tibetan and newar varieties. In *Proceedings of the workshop on resources and technologies for indigenous, endangered and lesser-resourced languages in eurasia within the 13th language resources and evaluation conference*, 1–6. Marseille, France: European Language Resources Association.

Faggionato, Christian & Marieke Meelen. 2019. Developing the Old Tibetan treebank. In *Proceedings of the international conference on recent advances in natural language processing (ranlp 2019)*, 304–312. Varna, Bulgaria: INCOMA Ltd. doi:10.26615/978-954-452-056-4_035.

Faggionato, Christian, Marieke Meelen & Nathan Hill. 2023. Classical Tibetan Annotation Manual Part II - Segmentation & POS tagging. doi:10.5281/zenodo.7880130.

Garrett, Edward John & Nathan W. Hill. 2015. Constituent order in the tibetan noun phrase, .

Guibon, Gaël, Marine Courtin, Kim Gerdes & Bruno Guillaume. 2020. When collaborative treebank curation meets graph grammars. In *Proceedings of the twelfth language resources and evaluation conference*, 5291–5300. Marseille, France: European Language Resources Association. https://aclanthology.org/2020.lrec-1.651.

de Jong, J. W. 1959. *Mi la ras pa' i rnam thar*. Berlin, Boston: De Gruyter Mouton. doi:10.1515/9783112313008.

Jurafsky, Daniel & James H Martin. 2014. *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Pearson Publishing.

Karlsson, Fred, Atro Voutilainen, Juha Heikkila & Arto Anttila (eds.). 1995. *Constraint Grammar: A Language-Independent System for Parsing Unrestricted Text*. Berlin: Mouton de Gruyter.

Matthews, Peter Hugoe et al. 1981. *Syntax*. Cambridge University Press.

Meelen, Marieke, Élie Roux & Nathan Hill. 2021. 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* 20(1). 1–11. doi:10.1145/3409488.

Nivre, Joakim, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty & Daniel Zeman. 2016. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the tenth international conference on language resources and evaluation (LREC'16)*, 1659–1666. Portorož, Slovenia: European Language Resources Association (ELRA). https://aclanthology.org/L16-1262.

Nivre, Joakim, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers & Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the twelfth language resources and evaluation conference*, 4034–4043. Marseille, France: European Language Resources Association. https://aclanthology.org/2020.lrec-1.497.

Straka, Milan, Jan Hajič & Jana Straková. 2016. UDPipe: Trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, POS tagging and parsing. In *Proceedings of the tenth international conference on language resources and evaluation (LREC'16)*, 4290–4297. Portorož, Slovenia: European Language Resources Association (ELRA). https://aclanthology.org/L16-1680.