Conversion from Pāṇinian Kārakas to Universal Dependencies for Hindi Dependency Treebank

Juhi Tandon, Himani Chaudhary, Riyaz Ahmad Bhat and **Dipti Misra Sharma**
Kohli Center on Intelligent Systems (KCIS),
International Institute of Information Technology,Hyderabad (IIIT-H)
Gachibowli,Hyderabad 500 032, India
{juhi.tandon,himani,riyaz.bhat}@research.iiit.ac.in , dipti@iiit.ac.in

Abstract

Universal Dependencies (UD) are gaining much attention of late for systematic evaluation of cross-lingual techniques for cross-lingual dependency parsing. In this paper we present our work in line with UD. Our contribution to this is manifold. We extend UD to Indian languages through conversion of Pāṇinian Dependencies to UD for the Hindi Dependency Treebank (HDTB). We discuss the differences in annotation in both the schemes, present parsing experiments for both the formalisms and empirically evaluate their weaknesses and strengths for Hindi. We produce an automatically converted Hindi Treebank conforming to the international standard UD scheme, making it useful as a resource for multilingual language technology.

1 Introduction

Universal Dependencies is a project undertaken to develop an inventory of languages that have treebanks annotated in a consistent scheme (McDonald et al., 2013). The UD annotation has evolved by reconstruction of the Standford Dependencies (De Marneffe and Manning, 2008) and it uses a slightly extended version of Google universal tag set for part of speech (POS) (Petrov et al., 2011). This is done with the motivation to facilitate the efforts in building of cross-linguistic tools such as parsers, translation systems, search engines, etc.

The efforts in building similarly structured or annotated treebanks have invoked a lot intreset from researchers around the world. The first release of UD treebanks included six languages where English and Swedish were created by automatic conversion. Thereafter several other treebanks have been developed automatically such as Italian (Bosco et al., 2013), Russian (Lipenkova and Soucek, 2014), and Finnish (Pyysalo et al., 2015). Several treebanks have also been created using manual annotation procedures. For languages where a treebank is already available, automatic conversion process is more suitable than manual annotation which is expensive and time consuming. It should be noted here that while for some languages conversion between the original and the UD representations can be accurate, for others it may introduce too much noise.

There have been few attempts that have tried to convert the annotation scheme used for Indian languages to other schemes such as the annotation style of Prague Dependency Treebank (Zeman et al., 2012). Our work, instead, aims to convert HDTB annotation scheme to UD.

Keeping in line with the ongoing efforts in this direction, our work is a volunteer effort to harmonize the Hindi Dependency Treebank according to the UD formalism, making it a more available resource for multilingual parsing. In doing so, we have converted the dependency relations in Pāṇinian framework and the POS tag set followed by Hindi to the Universal Dependency scheme. This conversion had its challenges, as many language specific phenomena had to be addressed in the process. However, there was no requirement to develop a new, language specific UD-scheme, unlike some other treebanks, for instance Russian (Lipenkova and Soucek, 2014).

The rest of the paper is organized as follows: In Section 2, we describe the annotation scheme used for the Indian language treebanking and Universal Dependency treebanking. Section 3 talks about the granularity of the Pāṇinian scheme. In Section 4, we elaborate upon the differences in design between the two schemes, how existing dependency scheme and POS tags for Hindi map onto the universal taxonomy, the issues that were faced

Proceedings of LAW X – The 10th Linguistic Annotation Workshop, pages 141–150,
Berlin, Germany, August 11, 2016.

Count of	HDTB
Types	22,171
Tokens	434,856
Chunks	233,864
Sentences	20,783
Avg. Tokens/Sentence	20.92
Avg. Chunks/Sentence	11.25

Count of	Training	Testing
Tokens	3,47,744	87,112
Chunks	1,87,029	46,835
Sentences	16,629	4,154

Table 1: General Treebank Statistics and training-testing split for all the experiments reported in this work.

and how they have been resolved. The conversion process and program is discussed in Section 5. Section 6 discusses the parsing performance of the two schemes, assesses the learnability of the automatic parser for the UD scheme and its suitability for Hindi. Lastly, we conclude and discuss future work in Section 7.

2 The Two Schemes

2.1 Hindi Dependency Treebank and Computational Paninian Grammar

The Hindi Treebank contains text from news articles and heritage domain. It consists of 434,856 tokens in 20,783 sentences with an average of 20.92 words per sentence as can be seen in Table 2. It is multi-layered and multi-representational (Bhatt et al., 2009; Xia et al., 2009; Palmer et al., 2009; Bhat et al., 2014). It contains three layers of annotation namely **dependency structure (DS)** for annotation of modified-modifier relations, **PropBank-style annotation** for predicate-argument structure, and an independently motivated **phrase-structure annotation**. Each layer has its own framework, annotation scheme, and detailed annotation guidelines.

Dependency Structure—the first layer in these treebanks—involves dependency analysis based on the Pāṇinian Grammatical framework (Bharati et al., 1995; Begum et al., 2008). Pāṇini was an Indian grammarian who is credited with writing a comprehensive grammar of Sanskrit. The underlying theory of his grammar provides a framework for the syntactico-semantic analysis of a sentence. The grammar treats a sentence as a series of modified-modifier relations where one of the elements (usually a verb) is the primary modified. This brings it close to a dependency analysis model as propounded in Tesnière's Dependency Grammar (Tesnière, 1959).

The syntactico-semantic relations between lexical items provided by the Pāṇinian grammatical model can be split into two types[1]:

1. **Kāraka**: These are semantically related to a verb as the direct participants in the action denoted by a verb root. The grammatical model has six 'kārakas', namely **'kartā'** (the doer), **'karma'** (the locus of action's result), **'karaṇa'** (instrument), **'sampradāna'** (recipient), **'apādāna'** (source), and **'adhikaraṇa'** (location). These relations provide crucial information about the main action stated in a sentence.

2. **Non-kāraka**: These relations include reason, purpose, possession, adjectival or adverbial modifications etc.

Both the **Kāraka** and **Non-kāraka** relations in the scheme are represented in Figure 1; glosses of these relations are given in Table 2 . The purpose of choosing a hierarchical model for relation types is to have the possibility of underspecifying certain relations.

2.2 Universal Dependencies

As mentioned by (Nivre et al., 2016) and also discussed by (Johannsen et al., 2015), the driving principles of UD formalism are:

1. **Content over function:** Content words form the backbone of the syntactic representation. Giving priority to dependency relations between content words increases the probability of finding parallel structures across languages, since function words in one language often correspond to morphological inflection (or nothing at all) in other languages. Functional heads are instead represented as specifying features of content words, using dedicated relation labels.

[1]The complete set of dependency relation types can be found in (Bharati et al., 2009)

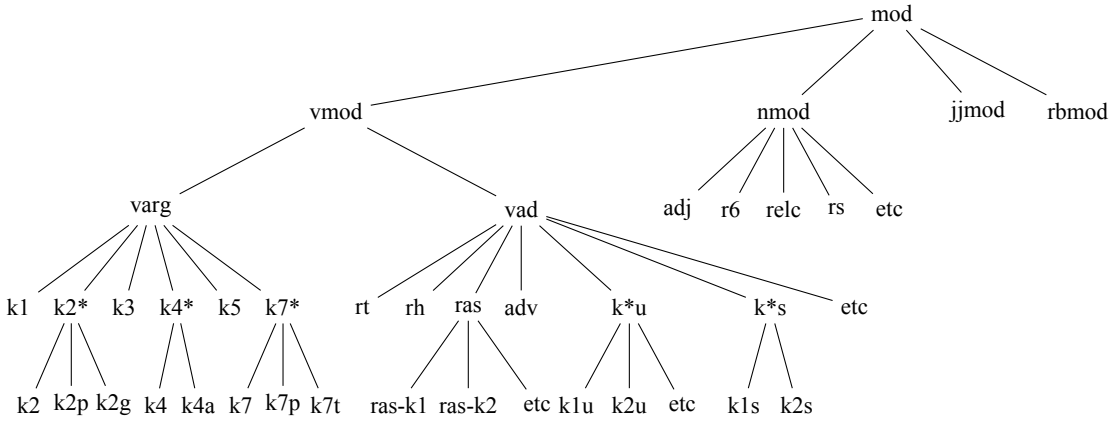

Figure 1: Inter-chunk dependency labels

Relation	Meaning
k1	Agent / Subject / Doer
k2*	Theme / Patient / Goal
k3	Instrument
k4*	Recipient / Experiencer
k5	Source
k7*	Spatio-temporal
rt	Purpose
rh	Cause
ras	Associative
k*u	Comparative
k*s	(Predicative) Noun / Adjective Complements
r6	Genitives
relc	Modification by Relative Clause
rs	Noun Complements (Appositive)
adv	Verb modifier
adj	Noun modifier

Table 2: Some major dependency relations depicted in Figure 1.

2. **Head-first:** In spans where it is not immediately clear which element is the head (the content-over-function rule does not apply straightforwardly), UD takes a head-first approach: the first element in the span becomes the head, and the rest of the span elements attach to it. This applies mostly to coordinations, multiword expressions, and proper names.

3. **Single root attachment:** There should be just one node with the root dependency relation in every tree, attached to the artificial root governor.

3 Granularity

Hindi is a morphologically rich, free word-order language. For such languages syntactic subject-object positions are not always able to elegantly explain the varied linguistic phenomena. As mentioned in the previous section, syntactico-semantic dependency relations and their labels defined in the CPG formalism are very fine grained to account for the rich grammatical functions. The number of distinct dependency labels are 82 as per the scheme (both interchunk and intrachunk). It has been observed that the more semantically oriented annotation schemes make labeled parsing more difficult than the schemes based on more surface-oriented grammatical functions. Further many applications do not require finer dependency labels and running a full parser with such a large set of labels can be too expensive. This further motivated us to convert the Hindi treebank to the UD scheme, with a relatively sparse taxonomy and observe the effects.

4 Differences in Design

While mapping the two annotation schemes we found that most of the tags entailed multiple correspondences, either one-to-many or many-to-one mappings between their tags. Below, some of the differences are discussed in detail.

4.1 Part of Speech Tags

The UD POS tag-set comprises 17 different tags only, whereas the POS tag set developed for Indian Languages (Bharati et al., 2006), has 32 tags.

One to many (HDTB to UD) The POS tag WQ used in the Hindi treebank for question words maps to DET, PRON and ADV in the UD.

Many to one (HDTB to UD) Similarly several tags on the Hindi treebank side RB, WQ etc. map with the UD POS tag ADV. Though we have a POS tag RB which directly maps with the grammatical category Adverb, our POS tagging scheme being more granular, we have various tags to annotate different kinds of adverbs. Tags such as WQ for question words ('kaha.N' कहाँ (where), 'kab' कब (when), 'kaise' कैसे (how)), NN (for words such as 'kal' कल (yesterday/tomorrow), 'Aj' आज (today)), NST, INTF, etc. are covered by UD under the umbrella tag ADV.

Hindi has compound conjunctions like 'aur to aur' और तो और (all the more) and 'jaisA ki' जैसा कि (like/as) etc. In HDTB these are tagged as follows:
और_CCC
तो_CCC
और_CC.
Multiword names are marked by POS tags NNPC and NNP. However in UD compounding is marked at the level of dependency relations by three tags: compound, mwe and name.

The Hindi tag set does not tag subordinate and coordinate conjunct differently. Our tags CC and CCC map with both, CONJ and SCONJ of UD for all simple and compound conjunctions.

There is not always a straight forward equivalence class mapping from HDTB to UD. The correct mapping of some tags requires the knowledge of lemma or the syntactic context. For example, the ambiguity in WQ and CC is resolved by using a word list corresponding to each Universal POS mapping and a few heuristics derived from the dependency tree structure.

In Indian Languages, there is a phenomenon called reduplication that involves the doubling of a lexical item to convey a grammatical function, such as plurality or intensification. The first word in such reduplicative construction is tagged by its respective lexical category and the second word is tagged as RDP to indicate that it is a case of reduplication, thus distinguishing it from a normal sequence (Bharati et al., 2006). UD does not have a corresponding tag for RDP which marks reduplication.

A mapping of all the 17 UD POS Tags to HDTB POS Tag set can be seen in the Table 3.

UD	HDTB
ADJ	JJ, JJC, QO
ADP	PSP, PSPC
ADV	RB, RBC, INTF, INTFC, NST, WQ, PRP, NN
AUX	VAUX, VAUXC
AUX	VAUXC
CONJ	CC, CCC
DET	DEM, QF, QFC, WQ
INTJ	INJ
NOUN	NN, NNC
NUM	QC, QCC
PART	RP, RPC, NEG
PART	NEG
PRON	PRP, PRPC, WQ
PROPN	NNP, NNPC
PUNCT	SYM
SCONJ	CC, CCC
VERB	VM, VMC
X	UNK

Table 3: Mappings of HDTB and Universal Dependencies POS tags.

4.2 Dependency Relations

In the above section, we found profound ambiguity in mapping the Hindi POS tags to their corresponding UD tags. In case of dependency relations, we also witnesses many cases of many-to-one and one-to-many mappings. For example dependency relations such as k3 (instrument of an action), k7t, k7p (location in time and space respectively), r6 (possession relation between two nouns), are all mapped to the label nmod (denoting nominal modifiers) in the UD scheme. Likewise, the Pāṇinian relation label k2 maps to xcomp, ccomp and dobj, based on 'Chunk[2] condition' described in section 5. The relation labels k1, k4a, pk1 (loosely corresponding to agent, experiencer, causer respectively) all map to the label nsubj of the UD scheme.

[2]A chunk (with boundaries marked), in HDTB, by definition, represents a group of adjacent words in a sentence, that are in dependency relation with each other, and where one of these words is their head (Chaudhry and Sharma, 2011)

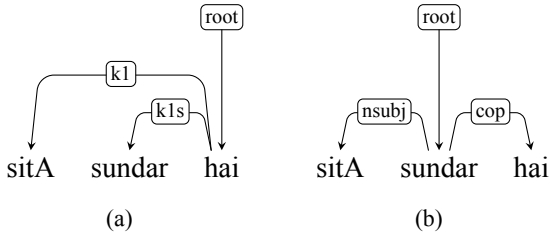

Figure 2: Dependency tree for a) HDTB and b) UD copula constructions.

4.3 Dependency Structure

In the Pāṇinian scheme there are about 82 kāraka and non kāraka relations. However, in UD there are only 40 dependency relations now, as opposed to 42 which were mentioned in (De Marneffe et al., 2014). The mapping between the two scheme can be seen in Table 4. One of the major challenges we came across during the conversion process was the conversion of elliptical constructions.

4.3.1 Copula

Currently in our scheme, a copular verb is considered as the head of a copula construction. During the conversion to UD, predicative nominal in the copula construction is marked as the head instead, while the 'be' verb becomes a `cop` dependent. For example in sentence (1) Sita is beautiful, 'sundar' सुंदर (beautiful) is treated as the head, while 'sItA' सीता (Sita) and the be verb 'hai' है (is) are its dependents of the type `nsubject` and `cop`, respectively.

(1) `सीता सुंदर है.'
'sItA sundar hai.'

sItA *sundar* *hai*
sita beautiful is
'Sita is beautiful.'

For conversion to UD, these relations must be reversed, not just relabeled, which in turn may cause structural changes of other kinds. For example, a reanalysis must be done for dependents of the previous governor and decision be made whether they should attach to the new governor or remain as they were. Thus, for conversion to UD these relations are reversed though it leads to structural changes as can be seen in Figure 2.

4.3.2 Conjunctions

Another type of constructions we handle are those with conjunctions. In HDTB annotation scheme a conjunction, either coordinating or subordinating is the head of the clause and the other elements of the clause are its dependents. In the sentence such as in Example (2), 'aur' (and) is annotated the head with 'rAm' (Ram), 'mohan' (Mohan), 'sItA' (Sita), and 'mIrA' (Meera) as its dependents.

(2) `राम, मोहन, सीता और मीरा आज आए थे.'
'rAm, mohan, sItA aur mIrA Aj Aye the.'

rAm, *mohAn,* *sItA* *aur* *mIrA* *Aj*
rAm mohAn sItA and mIrA today
Aye-the
came-PAST
'Ram, Mohan, Sita and Meera came today.'

Whereas in UD the first element of the coordinated construction is taken as the head. The conjunct and the other coordinated elements are annotated as its dependents. Given this sentence, in UD, 'rAm' is the head of the construction while 'mohan', 'sItA', 'mIrA' and 'aur' depend on it. Further, while 'mohan', 'sItA', 'mIrA' are annotated with the label `conj`, 'aur' is annotated with the label `CC`, since it is a coordinating conjunction, as can be seen in Figure 3. Also, UD annotates subordinating conjunction as `mark`, which is a dependent of the head of the subordinate clause. For the sake of conversion from HDTB to UD we distinguish between coordinating and subordinating conjunctions annotating them as `conj` and `mark`. For this we have enlisted them as two separate classes.

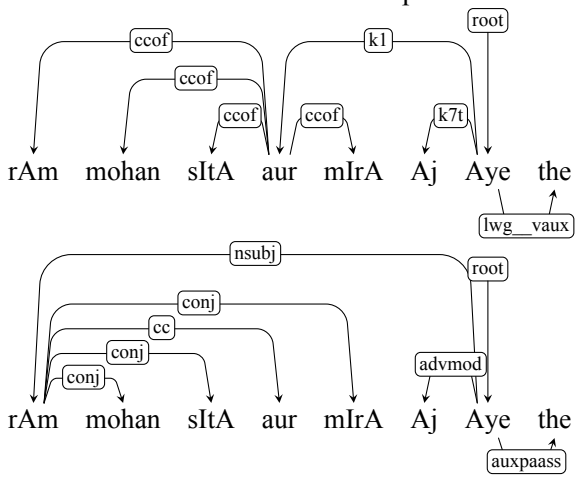

Figure 3: Dependency tree for a) HDTB and b) UD conjunctions constructions

4.3.3 Multiword names

As has been observed by (Johannsen et al., 2015), "In spans where it is not immediately clear which element is the head, UD takes a head-first approach: the first element in the span becomes the head, and the rest of the span elements attach to it. This applies mostly to coordinations, multiword expressions, and proper names." For example, in a name such as 'Atal Bihari Vajpayee', in UD, the first word in a compound name 'Atal', becomes the head and the rest its dependents. Whereas in HDTB, 'Vajpayee' is annotated the head and 'Atal' and 'Bihari' its dependents.

4.3.4 Ellipsis

Instances of ellipsis are abundant in the Hindi Treebank. While we are able to handle some in our current conversion, there are others we still need to work on. One such type which we have addressed is the 'yah-ki' यह-कि (this-that) complement constructions which follow the pattern: yah (*'यह)-its property-VP-ki* कि *clause'* (Mannem et al., 2009). In cases of ellipsis, a NULL node is introduced to facilitate annotation, since the entire 'ki' (that) clause is annotated as the child of 'yaha' (this) /`NULL' node here.

In Hindi, sentences such as in Example (3):

(3) `गौरतलब है कि गोपाल को नासा आमंत्रित किया गया था.'
'gaurtalab hai ki gopAl ko nAsA Amantrit kiyA gayA thA.'

gaurtalab hai ki gopAl-ko nAsA
to-be-noted is that Gopal to-NASA
Amantrit-kiyA-gayA-thA.
invited-was
'Is to be noted that Gopal was invited to NASA.

'gaurtalab hai ki gopAl ko nAsA Amantrit kiyA gayA thA' (Is to be noted that Gopal was invited to NASA.) can come with the referent 'yah' यह (this) elided (a case of Pronoun drop) or explicitly manifested in the sentence. The 'ki' कि (that) clause and its referent are both modifiers (child) of the verb. However, in HDTB annotation the 'ki' clause is annotated a modifier of its referent which in turn is marked as the child of the verb. For the sake of consistency in cases where 'yah', the referent, does not manifest explicitly, a 'NULL' node is introduced in its stead. However, the UD scheme does not introduce NULL tokens to represent elided elements. Therefore to map all 'ki' complement constructions, with the UD scheme, we drop the 'NULL' node, and the 'ki' complement clause is annotated a dependent of the head of the removed `NULL' node (usually the verb) as illustrated in Figure 4.

Universal	Pāṇinian
acl	nmod__k1inv, nmod__k2inv, nmod__relc, rs, k2g, k2s, rbmod__relc
neg	nmod__neg
dislocated	fragof
iobj	k4
nmod	k2u, jk1, k1u, k3, k3u, k2p, k4u, k5, k7, k7a, k7p, k7pu, k7t, k7tu, k7u, r6, r6-k1, r6-k2, r6v, ras-k1, ras-k1u, ras-k2, ras-k4, ras-k4a, ras-k7, ras-k7p, ras-neg, ras-pof, ras-r6, ras-r6-k2, ras-rt, nmod__emph
punct	rsym
vocative	rad
advmod	rd, rsp, lwg__intf, vmod__adv, jjmod__intf, jjmod, adv, rbmod
dep	lwg__rp, lwg__unk, undef, enm
compound	pof__cn, pof__redup, lwg__rdp, lwg__vm, nmod__pofinv, pof, pof__inv
case	lwg__psp, lwg__nst, psp__cl
neg	lwg__neg
det	mod__wq
dobj	k2, k1s, mk1
amod	nmod__adj, nmod
parataxis	vmod
ccomp	k2
xcomp	k2
aux	lwg__vaux
auxpass	lwg__vaux
nsubj	k1, k4a, pk1
nsubjpass	k1
advcl	rh, rt, rtu, sent-adv, vmod

Table 4: Universal mapping of Pāṇinian Dependencies used in HDTB.

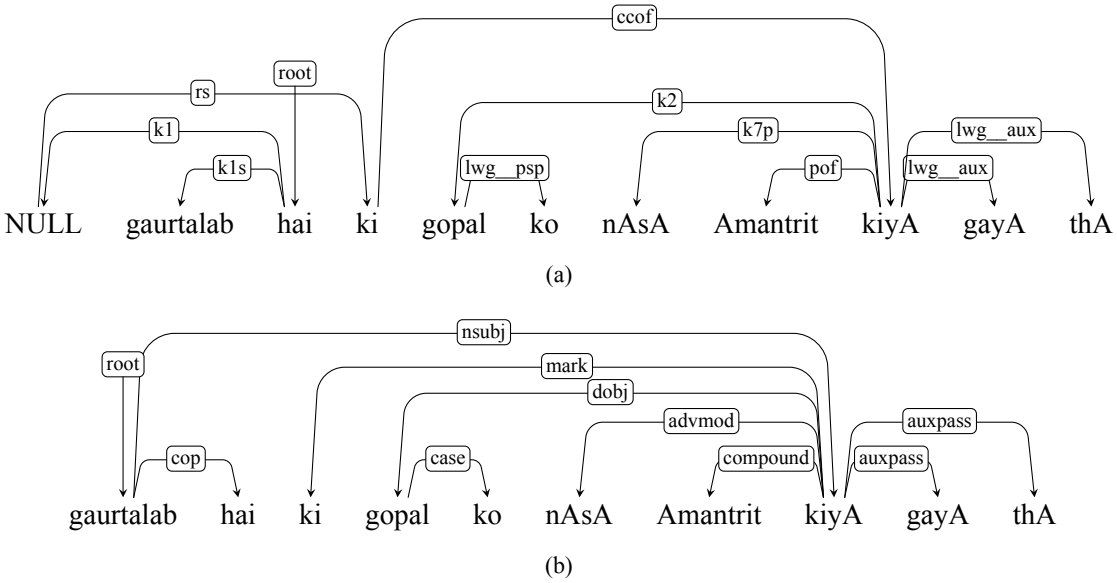

Figure 4: Dependency tree for a) HDTB and b) UD ellipsis constructions.

5 Conversion Process

During conversion it must be noted that we are moving from a very detailed and granular format to a format which is under-specified. The implementation of the conversion was based on the mapping between schemes described above. The conversion script executes as a pipeline of three components, each of which takes as input data in CONLL format and outputs data in the same format. During conversion, structure is handled before labels. The first module harmonizes the structural differences from HDTB to UD by handling ellipsis (and thereafter aligning nodes in the tree after NULL removal), copula constructions, mutliword names and conjunctions. It updates the nodes as modifier-modified relations have been changed. The second and third module converts POS and Dependency relations from HDTB to UD, respectively. The conversion is based on certain heuristics which involve conditions specified in terms of lexical, structural, morphological information and Part of Speech tags. Examples for the different types of conversion conditions are as follows:

- Lexical condition: The POS tag `WQ` of HDTB is converted to `ADV` of UD when expressed by word form or lemma 'kab' (कब), 'kahA.N' (कहाँ), 'kaisA' (कैसा), 'kyun' (क्यों). Else if the node has chunk type as child in its features, it is converted to `DET` of UD; otherwise to `PRON` of UD.
- Morphological condition: If the dependency relation is any of `k1`, `pk1`, `k4a` and the current node's parent has TAM (Tense, Aspect, Modality) feature as *'yA_jA'* (या_जा), the relation is converted to `nsubjpass`; if dependency relation is `lwg__vaux` and there is a presence of the TAM feature, it is converted to `auxpass`. In the absence of this morph feature `lwg__vaux` is converted to `aux`, while `k1`, `pk1`, `k4a` are converted to `nsubj`.
- Chunk condition: If the dependency relation is `k2` and the current node's chunk id is `VGF` (Finite Verb Chunk), the relation is converted to `ccomp`; else if chunk id is `VGNN` (Verb Chunk - Gerund) it is converted to `xcomp`; otherwise to `dobj`.
- POS condition: If the dependency relation is any of `nmod__adj` or `nmod` and the node's POS Tag is `DET` or `DEM` the relation is converted to `det`; if POS is `NUM` it is converted to `nummod`; else if POS is any of `NNP`, `NNPC`, `PRP`, `NN` or `NNC`, it is converted to `nmod`.

During conversion from HDTB to UD we lose 3852 sentences that cannot be restructured according to our current scheme, they are mostly cases of ellipsis (gapping).

6 Parsing Experiment

Our motive of conversion from HDTB to UD was not keeping the increase/decrease of parsing accu-

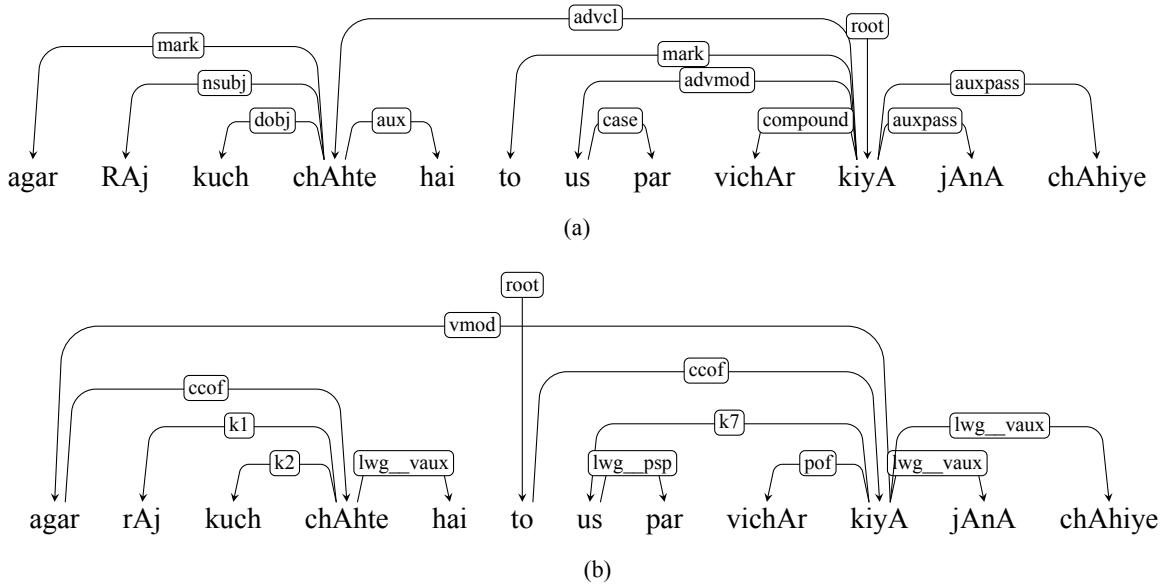

Figure 5: Dependency tree for paired connective 'agar-to' for a) UD and b) HDTB.

racy in consideration. The design choices taken, such as the head-first approach, as described in Section 2.2, led to changes in a lot of attachments like that of Copula and decreased the learnability of the parser for the syntactic structure of Hindi. We conduct parsing experiments to quantify these effects and also do a manual error analysis to point out the constructions which are not learnt efficiently by the parser.

For experiment purposes, we are using MALT [3] with parser settings from (Ambati et al., 2010). The metrics used for evaluation are Labeled and Unlabeled attachment score (LAS, UAS) and Label accuracy score (LS). The average accuracy of 10 fold cross validation is reported in Table 5.

We experience an accuracy drop of ∼2% in UAS in conversion from HDTB to UD. This is not surprising as the two are now quite different treebanks. The drop in accuracy can be attributed to the numerous changes in attachment of edges while conversion. However the increase in accuracy of LS is intuitive because of the reduced number of classes of classification for dependency relations.

On doing a manual error analysis we observe the following patterns:

- The parser is not able to learn copula constructions properly, the 'be' verb is not recognised as 'cop' in most cases, it is made the root of the sentence or head of the phrase, instead. This is at odds with the general structure where verb is a root of a dependency tree as it is the primary modified. These structures also cannot be learnt efficiently based on lexicalism as the 'be' verb is also used as an auxillary in most cases.
- For control constructions which have more than one verb, the first non-finite verb is marked as the head instead of the finite verb.
- Sentences having paired connectives like 'agar-to' (अगर-तो), 'yadi-to' (यदि-तो) corresponding to 'if-then', do not have their governors and dependents correctly marked. This is because they are handled differently in both the schemes. In HDTB 'agar'and 'to' are clausal heads. The 'to'-clause is modified by the 'agar'-clause. Whereas in UD 'agar'and 'to' must be dependents of the main verb of their respective clauses as can be seen in Figure 5.

7 Conclusion and Future Work

In this paper we briefly described the process of conversion of Hindi Treebank to UD annotation scheme. It was an attempt to release the resource in a widely accepted international format so that it becomes more usable for a variety of multilingual NLP tasks. The conversion was a challenging task and there are constructions which are yet to be addressed to be fully compliant with the UD scheme

[3] MALT version 1.8.1 , http://www.maltparser.org/

	LAS	UAS	LS
Pāṇinian	90.97	95.206	92.908
UD	90.237	93.193	94.053

Table 5: Average accuracy of 10-fold cross validation using Pāṇinian and UD framework.

like that of ellipsis etc. As a part of future work, we propose to come up with better techniques to resolve empty nodes in the absence of predicational or verbal heads. Also a few attachment schemes must be reanalyzed and revised to handle long distance dependencies efficiently. We performed experiments using MALT parser on both the source treebank HDTB and the converted UD Hindi Treebank, to find that the performance is slightly deteriorated after conversion.

Acknowledgments

We would like to thank anonymous reviewers for their valuable comments that helped to improve the quality of this paper.
The work reported in this paper is supported by the NSF grant (Award Number: CNS 0751202; CFDA Number: 47.070)[4]

References

Bharat Ram Ambati, Samar Husain, Joakim Nivre, and Rajeev Sangal. 2010. On the role of morphosyntactic features in hindi dependency parsing. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 94–102. Association for Computational Linguistics.

Rafiya Begum, Samar Husain, Arun Dhwaj, Dipti Misra Sharma, Lakshmi Bai, and Rajeev Sangal. 2008. Dependency annotation scheme for indian languages. In *IJCNLP*, pages 721–726. Citeseer.

A. Bharati, V. Chaitanya, R. Sangal, and KV Ramakrishnamacharyulu. 1995. *Natural Language Processing: A Paninian Perspective*. Prentice-Hall of India.

Akshar Bharati, Rajeev Sangal, Dipti Misra Sharma, and Lakshmi Bai. 2006. Anncorra: Annotating corpora guidelines for pos and chunk annotation for indian languages. *LTRC-TR31*.

[4]Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Akshar Bharati, DM Sharma S Husain, L Bai, R Begam, and R Sangal. 2009. Anncorra: Treebanks for indian languages, guidelines for annotating hindi treebank (version–2.0).

Riyaz Ahmad Bhat, Rajesh Bhatt, Annahita Farudi, Prescott Klassen, Bhuvana Narasimhan, Martha Palmer, Owen Rambow, Dipti Misra Sharma, Ashwini Vaidya, Sri Ramagurumurthy Vishnu, et al. 2014. The hindi/urdu treebank project. In *Handbook of Linguistic Annotation*. Springer Press.

Rajesh Bhatt, Bhuvana Narasimhan, Martha Palmer, Owen Rambow, Dipti Misra Sharma, and Fei Xia. 2009. A multi-representational and multi-layered treebank for hindi/urdu. In *Proceedings of the Third Linguistic Annotation Workshop*, pages 186–189. Association for Computational Linguistics.

Cristina Bosco, Simonetta Montemagni, and Maria Simi. 2013. Converting italian treebanks: Towards an italian stanford dependency treebank. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 61–69. Citeseer.

Himani Chaudhry and Dipti M Sharma. 2011. Annotation and issues in building an english dependency treebank.

Marie-Catherine De Marneffe and Christopher D Manning. 2008. The stanford typed dependencies representation. In *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8. Association for Computational Linguistics.

Marie-Catherine De Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D Manning. 2014. Universal stanford dependencies: A cross-linguistic typology. In *LREC*, volume 14, pages 4585–4592.

Anders Johannsen, Héctor Martínez Alonso, and Barbara Plank. 2015. Universal dependencies for danish. In *International Workshop on Treebanks and Linguistic Theories (TLT14)*, page 157.

Janna Lipenkova and Milan Soucek. 2014. Converting russian dependency treebank to stanford typed dependencies representation. In *EACL*, pages 143–147.

Prashanth Mannem, Himani Chaudhry, and Akshar Bharati. 2009. Insights into non-projectivity in hindi. In *Proceedings of the ACL-IJCNLP 2009 Student Research Workshop*, pages 10–17. Association for Computational Linguistics.

Ryan T McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith B Hall, Slav Petrov, Hao Zhang, Oscar Täckström, et al. 2013. Universal dependency annotation for multilingual parsing. In *ACL (2)*, pages 92–97. Citeseer.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, et al. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*.

Martha Palmer, Rajesh Bhatt, Bhuvana Narasimhan, Owen Rambow, Dipti Misra Sharma, and Fei Xia. 2009. Hindi syntax: Annotating dependency, lexical predicate-argument structure, and phrase structure. In *The 7th International Conference on Natural Language Processing*, pages 14–17.

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2011. A universal part-of-speech tagset. *arXiv preprint arXiv:1104.2086*.

Sampo Pyysalo, Jenna Kanerva, Anna Missilä, Veronika Laippala, and Filip Ginter. 2015. Universal dependencies for finnish. In *Proceedings of NoDaLiDa*, pages 163–172.

Lucien Tesnière. 1959. *Eléments de syntaxe structurale*. Librairie C. Klincksieck.

Fei Xia, Owen Rambow, Rajesh Bhatt, Martha Palmer, and Dipti Misra Sharma. 2009. Towards a multi-representational treebank. In *The 7th International Workshop on Treebanks and Linguistic Theories. Groningen, Netherlands*, pages 159–170.

Daniel Zeman, David Marecek, Martin Popel, Loganathan Ramasamy, Jan Stepánek, Zdenek Zabokrtskỳ, and Jan Hajic. 2012. Hamledt: To parse or not to parse? In *LREC*, pages 2735–2741.