A Treebank-based Investigation of IPP-triggering Verbs in Dutch

Liesbeth Augustinus
liesbeth@ccl.kuleuven.be

Frank Van Eynde
frank@ccl.kuleuven.be

Centre for Computational Linguistics, KU Leuven

Abstract

Based on a division into subject raising, subject control, and object raising verbs, syntactic and semantic distinctions are drawn to account for the differences and similarities between verbs triggering *Infinitivus Pro Participio* (IPP) in Dutch. Furthermore, quantitative information provides a general idea of the frequency of IPP-triggering verb patterns, as well as a more detailed account of verbs which optionally trigger IPP. The classification is based on IPP-triggers occurring in two treebanks.

1 Introduction

Infinitivus Pro Participio (IPP) or *Ersatzinfinitiv* appears in a subset of the West-Germanic languages, such as German, Afrikaans, and Dutch. Only verbs that select a (*te*-)infinitive are possible IPP-triggers. If those verbs occur in the perfect tense, they normally appear as a past participle, such as *gevraagd* 'asked' in (1). In IPP constructions, however, those verbs appear as an infinitival verb form instead of a past participle, such as *kunnen* 'can' in (2). Haeseryn et al. [3], Rutten [5], and Schmid [7] (among others) provide lists of verbs which trigger IPP. The examples show that for some verbs, IPP is triggered obligatorily (2), whereas for other verbs, IPP appears optionally (3a - 3b), or is ungrammatical (1).

(1) Of bent u als partner **gevraagd** deel te nemen in een IST project ...?
Or are you as partner ask-PSP part to take in an IST project
'Or are you asked as a partner to take part in an IST project ...?' (WR-P-E-E-0000000019.p.2.s.2)

(2) Pas nu hebben we dat ook **kunnen** zien in de hersenen.
Only now have we that also can-INF see in the brains
'Only now we have been able to see that in the brains.' (dpc-ind-001634-nl-sen.p.16.s.5)

(3) a. ik heb m'n nicht **proberen** te bellen want die uh ...
I have my cousin try-INF to call because that one uh
'I have tried to call my cousin because she uh ...' (fna000628__73)

b. daar hebben we toen **geprobeerd** te bellen ...
there have we then try-PSP to call
'Then we have tried to call there' (fna000260__277)

By means of a treebank-based investigation this paper aims to set up a typology of Dutch IPP-triggering verbs. The results are used to verify the existing descriptive approaches as well as to gain more insight in (the frequency of) the phenomenon. Furthermore, the results turn out to be useful for the creation and optimization of NLP applications, since most verbs under investigation also have a main verb function besides their verb-selecting function.

We start from the division into subject raising, object raising, subject control, and object control verbs, following Sag et al. [6]. *Subject raising verbs*, such as *kunnen* 'can' in (2), do not assign a semantic role to their subject, whereas *subject control verbs*, such as *proberen* 'try' in (3), do.[1] Similarly, *object raising verbs*, such as the perception verb *zien* 'see', do not assign a semantic role to their object, whereas *object control verbs*, such as *vragen* 'ask' in (1), do.

2 Data and Methodology

In order to get a data-driven classification of IPP-triggers, we have explored two manually corrected treebanks: LASSY Small (van Noord et al. [8]) for written Dutch, and the syntactically annotated part of CGN (Hoekstra et al. [4]) for spoken Dutch. Each of those treebanks contains ca. one million tokens. Using treebanks instead of 'flat' corpora is interesting for this topic, since it is possible to look for syntactic dependency relations without defining a specific word order. In main clauses the finite verb and the final verb cluster are not necessarily adjacent, which makes such constructions hard to retrieve using string-based search.

Both treebanks can be queried with XPath, a W3C standard for querying XML trees.[2] The tools used for treebank mining are GrETEL (Augustinus et al. [1]) and Dact (de Kok [2]).[3] Exploratory research, such as finding relevant constructions and building general queries (XPath expressions) was done by means of GrETEL, a user-friendly query engine. In a following step, Dact was used to refine the queries and to examine the data in detail.

Figure 1 presents two general queries. In order to retrieve IPP-triggers, we looked for constructions with a form of *hebben* or *zijn* (perfect tense auxiliaries) as head (HD) that have an infinitival verbal complement (VC) as sister node. That VC node contains the IPP-trigger as well as another VC, which can take on the form of a bare infinitive or a *te*-infinitive, cf. Figure 1a. We also looked for similar constructions without IPP, i.e. constructions with an auxiliary of the perfect, a past

[1] One way to differentiate subject raising from subject control verbs is testing whether the verbs can combine with non-referential subjects (e.g. *Het begint te regenen* 'It starts to rain' vs. **Het probeert te regenen* '*It tries to rain').

[2] http://www.w3.org/TR/xpath

[3] http://nederbooms.ccl.kuleuven.be/eng/gretel; http://rug-compling.github.com/dact

participle and a (*te*-)infinitival complement (further referred to as: 'PSP constructions'), cf. Figure 1b.

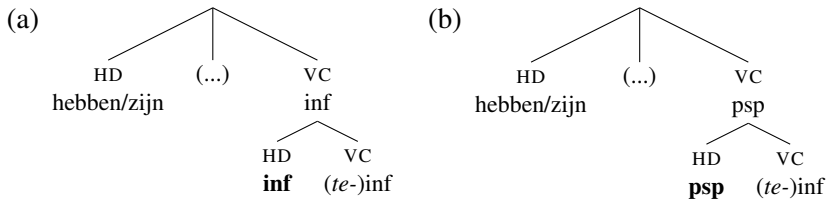


Figure 1: Query trees for IPP (a) and PSP (b) constructions

After extracting the relevant constructions from the corpus, the IPP-triggers were divided into the verb classes introduced in section 1. The resulting typology will be discussed in section 3.

3 Results

We have found 16.510 occurrences of verbs taking a (*te*-)infinitival complement in LASSY, and 20.729 in CGN. Since IPP-triggers can only be detected in constructions with a perfective auxiliary, only a small subset of those corpus hits can be used for our study. The IPP-triggers account for 1.9% of the verbs taking a (*te*-)infinitival complement in LASSY, and 3.7% in CGN. Still, IPP turns out to be a common phenomenon in a small set of frequently occurring verbs. Table 1 presents the general results that were found by means of the queries formulated in Figure 1. Those results reveal that in both treebanks IPP constructions occur more often than PSP constructions. In CGN, the IPP-triggers account for 96.1% of the constructions under investigation.

	LASSY		CGN	
	# hits	% hits	# hits	% hits
IPP	310	70.9	771	96.1
PSP	127	29.1	31	3.9
TOTAL	**437**	**100**	**802**	**100**

Table 1: General search results in CGN and LASSY Small

Based on the criteria formulated in Sag et al. [6], we have divided the verbs in three classes: *subject raising*, *object raising*, and *subject control* verbs. No object control verbs (e.g. *overtuigen* 'persuade') have been detected as IPP-triggers.

We only considered (*te*-)infinitive selecting verbs that allow perfect tense constructions. An example of a verb that does not allow the perfect (and therefore no IPP nor PSP) is *plegen* 'be in the habit of, tend' in (4).

(4) Zoals oud-collega Jennes Becheld 't **placht** te zeggen.
As former collegue Jennes Becheld it tend to say
'As former collegue Jennes Becheld tends to say.' (fna000432__125)

Table 2 presents the list of Dutch IPP-triggers.[4] For each lemma, we have indicated how many times it appears as the selector of a (*te*-)infinitival complement (VC[inf]-sel), the frequency of its occurrence in IPP and PSP constructions, as well as its semantic classification (if applicable).[5]

	LASSY			CGN				
Lemma	**VC[inf]-sel**	**IPP**	**PSP**	**VC[inf]-sel**	**IPP**	**PSP**	**Translation**	**Subtype**
subject raising verbs								
durven	0	0	0	1	0	0	'be on the point of' (literally: 'dare')	
beginnen	132	4	3	163	8	4	'start, begin'	aspectual
blijven	294	9	0	170	20	0	'continue'	aspectual
gaan	529	28	0	2715	157	0	'go, will'	aspectual
ophouden	0	0	0	0	0	0	'stop'	aspectual
behoeven	3	0	0	1	0	0	'need'	modal
(be)horen	5	0	0	15	0	0	'ought to'	modal
dienen	164	1	0	13	0	0	'have to'	modal
hoeven	121	2	0	204	5	0	'have to'	modal
kunnen	4154	72	0	4190	139	0	'can'	modal
moeten	2513	57	0	4272	128	0	'have to'	modal
mogen	624	6	0	664	22	0	'may'	modal
zullen	3274	0	-	3076	0	-	'will'	modal
blijken	167	0	0	61	0	0	'turn out'	evidential
lijken	124	0	0	52	0	0	'seem'	evidential
schijnen	11	0	0	38	0	0	'appear'	evidential
object raising verbs								
weten	0	0	0	0	0	0	'know, remember'	
helpen	41	1	0	7	0	0	'help'	benefactive
leren	1	0	0	2	2	0	'teach'	benefactive
doen	87	2	0	60	4	0	'do'	causative
laten	566	68	0	610	86	0	'let'	causative
horen	8	2	0	40	16	0	'hear'	perception
voelen	3	0	0	8	0	0	'feel'	perception
zien	61	8	0	167	27	0	'see'	perception
subject control verbs								
vermogen	0	0	0	0	0	0	'be able to'	
dreigen	16	0	4	3	0	2	'threaten'	
hangen	0	0	0	0	0	0	'hang'	aspectual
komen	127	8	0	214	18	0	'come'	aspectual
liggen	5	0	0	23	0	0	'lay'	aspectual
lopen	7	0	0	20	5	0	'walk'	aspectual
staan	22	4	0	99	11	0	'stand'	aspectual
zijn (wezen)	0	0	0	12	12	0	'be in the activity of'	aspectual
zitten	18	2	0	351	40	0	'sit'	aspectual
leren	40	2	1	46	16	0	'learn'	commitment
pogen	8	0	1	0	0	0	'try'	commitment
proberen	230	1	17	265	8	10	'try'	commitment
trachten	29	1	1	12	0	0	'try'	commitment
weigeren	66	0	0	5	0	0	'refuse'	commitment
weten	117	13	0	38	5	0	'know (how to)'	commitment
zien	5	0	0	14	0	0	'intend'	commitment
zoeken	1	0	0	0	0	0	'intend'	commitment
believen	0	0	0	0	0	0	'please'	mental orientation
durven	29	3	0	56	5	0	'dare'	mental orientation
menen	9	0	0	10	0	0	'mean, intend'	mental orientation
willen	1112	16	0	1476	37	0	'want'	mental orientation
TOTAL	**14723**	**310**	**27**	**19173**	**771**	**16**		

Table 2: IPPs in CGN and LASSY Small

[4]Due to limitations of presentation, we have only provided a list of the IPP-triggers. Verbs that occur in PSP constructions but not as IPP-triggers are not presented.

[5]PSP constructions with *zullen* 'will' do not exist, because *zullen* does not have a past participle. This is indicated in the table with a dash (-).

Some verbs occur twice, since they have different meanings. For example, the verb *leren* is a raising verb if it has the meaning 'teach' (5a), but it is a control verb if it denotes the meaning 'learn' (5b).

(5) a. 'k heb mijn kleine kinderen daar ook **leren** zwemmen .
I have my small children there also teach-INF swim
'I have also taught my little children how to swim over there' (fva400659__44)

b. In 2001 heb ik saxofoon **leren** spelen .
in 2001 have I saxophone learn-INF play
'In 2001 I learned to play the saxophone.' (dpc-qty-000936-nl-sen.p.36.s.2)

The results of the treebank mining reveal that the raising verbs (that allow perfective constructions), allow IPP. The modal, evidential, causative, and perception verbs all obligatorily trigger IPP. Raising verbs that optionally trigger IPP are the benefactive object raisers *leren* 'teach' and *helpen* 'help', and the aspectual subject raisers *beginnen* and *ophouden* 'stop'.

The most heterogenous group consists of subject control verbs, as that category contains verbs that obligatorily trigger IPP (e.g. aspectual motion and position verbs such as *lopen* 'walk' and *staan* 'stand', cf. (6)), verbs that do not allow IPP (e.g. *besluiten* 'decide', cf. (1)), and verbs which occur in both IPP and PSP constructions (e.g. *proberen* 'try', cf. (3a-3b)).

(6) Met wat meer geluk hadden we hier **staan** juichen.
With some more luck had we here stand-INF cheer
'With a bit more luck we would have been cheering.' (WR-P-P-H-0000000020.-p.14.s.8)

Due to data sparseness, not all IPP-triggers occur in the treebanks. In order to get a more complete list, we have also included verbs which occur as IPP-trigger on the Internet using Google search.[6] An example is the aspectual subject raiser *ophouden* 'stop', cf. (7). The frequency information of those Google results is not included in Table 2.

(7) Een reisorganisatie die helaas heeft **ophouden** te bestaan.
A travel organisation that unfortunately has stop-INF to exist
'A travel company that unfortunately has ceased to exist.' (Google, 30-10-2012)

In order to overcome the problem of data sparseness, future research will focus on collecting IPP-triggers in larger treebanks (e.g. LASSY Large, which currently contains 2.4 billion tokens). If more data are found, we will also be able to investigate in which situation the PSP construction is preferred over an IPP construction.

[6] www.google.be/www.google.nl

4 Conclusions and Future Research

Starting from a division into subject raising, subject control, and object raising verbs (Sag et al. [6]), we made a distinction along syntactic and semantic lines to account for the differences and similarities between IPP-triggers, which cannot be derived from the list provided by Haeseryn et al. [3].

The classification is supplemented with quantitative information to provide a general idea of the frequency of IPP-triggering verbs on the one hand, as well as a more detailed account of verbs which optionally trigger IPP on the other hand. On a global level, IPP seems to be more common in spoken than in written Dutch. The classification furthermore shows that subject and object raising verbs always allow IPP constructions if constructions with a perfective auxiliary are allowed. Object control verbs never occur as IPP-triggers, whereas the subject control verbs can be subdivided into verbs that obligatorily trigger IPP, optionally trigger IPP, or cannot occur as IPP.

In future research, we will investigate whether our classification works for other languages that have IPP-triggers, such as German and Afrikaans.

References

[1] L. Augustinus, V. Vandeghinste, and F. Van Eynde. Example-Based Treebank Querying. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC-2012)*, Istanbul, 2012.

[2] D. de Kok. Dact [Decaffeinated Alpino Corpus Tool].
URL: http://rug-compling.github.com/dact.

[3] W. Haeseryn, K. Romijn, G. Geerts, J. de Rooij, and M. van den Toorn. *Algemene Nederlandse Spraakkunst*. Martinus Nijhoff/Wolters Plantyn, Groningen/Deurne, second edition, 1997.

[4] H. Hoekstra, M. Moortgat, B. Renmans, M. Schouppe, I. Schuurman, and T. van der Wouden. *CGN Syntactische Annotatie*, 2003. 77p.

[5] J. Rutten. *Infinitival Complements and Auxiliaries.* PhD thesis, University of Amsterdam, 1991.

[6] I. Sag, T. Wasow, and E. Bender. *Syntactic Theory. A Formal Introduction.* CSLI Publications, Stanford, second edition, 2003.

[7] T. Schmid. *Infinitival Syntax. Infinitivus Pro Participio as a Repair Strategy.* John Benjamins, Amsterdam/Philadelphia, 2005.

[8] G. van Noord, G. Bouma, F. Van Eynde, D. de Kok, J. van der Linde, I. Schuurman, E. Tjong Kim Sang, and V. Vandeghinste. Large Scale Syntactic Annotation of Written Dutch: Lassy. In *Essential Speech and Language Technology for Dutch: resources, tools and applications*. Springer, in press.

Profiling Feature Selection for Named Entity Classification in the TüBa-D/Z Treebank

Kathrin Beck, Erhard Hinrichs

Department of General and Computational Linguistics
University of Tübingen
Email: {kbeck;eh}@sfs.uni-tuebingen.de

Abstract

This paper provides an overview of the named entity annotation included in the TüBa-D/Z treebank of German newspaper articles. It describes the subclasses of named entities distinguished by the TüBa-D/Z annotation scheme and profiles a set of surface-oriented and syntax-based features that are highly predictive for different subclasses of named entities.

1 Introduction

The annotation and automatic detection of named entities (henceforth abbreviated as NE) in large corpora has played a major role in recent research in computational linguistics and in the field of digital humanities. In digital humanities research, NEs have featured prominently in the creation of linked data for classical text collections and the visualization of the content of corpus collections by linking NEs with their geospatial coordinates. In computational linguistics, the identification of NEs is an important ingredient in the automatic detection of coreference relations in texts, of automatic topic detection and text classification, as well as for question-answering and other information retrieval and extraction applications.

In order to provide training material for (semi-)supervised learning algorithms for automatic NE detection, the annotation of corpus materials with NEs has been an important desideratum. For this reason, the linguistic annotation of the Tübingen Treebank of Written German (TüBa-D/Z)[1] has been enhanced in recent years by NE information. To the best of our knowledge, the TüBa-D/Z NE annotation constitutes the largest dataset of this kind for German apart from the German data prepared for the CoNLL-2003 NE recognition shared task (Tjong Kim Sang and de Meulder [9]) and the GerNED corpus (Ploch et al. [6]) (cf. Table 1).

[1] The TüBa-D/Z Treebank (PID: http://hdl.handle.net/11858/00-1778-0000-0005-896C-F) is freely available on http://www.sfs.uni-tuebingen.de/en/ascl/resources/corpora/tuebadz/.

The purpose of the present paper is twofold:

(i) To provide an overview of the classification scheme and the annotation principles used in NE annotation in the TüBa-D/Z, and

(ii) To identify those linguistic features that show a high correlation with particular subclasses of NEs distinguished in the TüBa-D/Z. Such linguistic profiling of different types of NEs is of immediate value for applications in computational linguistics. It can inform feature selection for machine learning approaches to the automatic detection and classification of NEs. It can also help define relevant linguistic patterns for unsupervised data mining techniques for NE extraction from very large corpora, including web-harvested data.

The remainder of the paper is structured as follows: Section 2 provides a brief overview of the linguistic annotation levels present in the TüBa-D/Z. Section 3 introduces TüBa-D/Z's NE classification and the linguistic criteria that inform NE classification in the TüBa-D/Z. Section 4 describes the overall distribution of the five different subclasses of NE found in the TüBa-D/Z data and profiles two types of linguistic features that show a high correlation with particular subclasses of NEs: word-based co-occurrences (subsection 4.1) and syntax-based features (subsection 4.2). Section 5 summarizes the findings of the paper and discusses some directions for future research.

2 TüBa-D/Z Overview

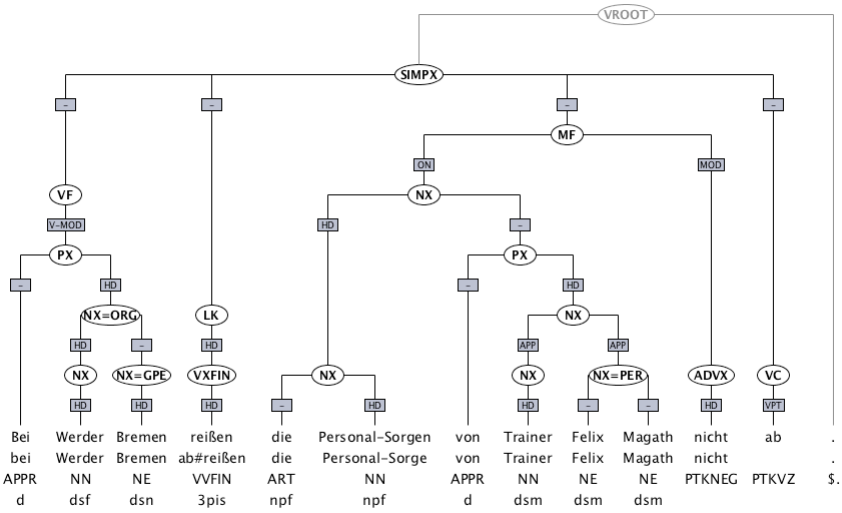
The TüBa-D/Z treebank is a syntactically annotated German newspaper corpus based on data taken from the daily issues of 'die tageszeitung' (taz). The treebank currently comprises 65,524 sentences (1,164,766 tokens).

Figure 1 illustrates the annotation layers of part-of-speech annotation, syntactic annotation, and NE annotation. The terminal nodes in the parse tree are labeled with part of speech tags taken from the Stuttgart Tübingen Tagset [7]. Non-terminal nodes of the tree include maximal projections of lexical categories such as ADVX (adverbial phrase), NX (noun phrase), and PX (prepositional phrase), but also a layer of topological fields such as VF (Vorfeld), LK (Linke Klammer), MF (Mittelfeld), and VC (Verbkomplex). Topological fields in the sense of Höhle [5], Herling [4], and Drach [1] are widely used in descriptive studies of German syntax. Such fields constitute an intermediate layer of analysis above the level of individual phrases and below the clause level.

Apart from syntactic phrase labels and topological fields, the syntactic annotation layer of the TüBa-D/Z also includes grammatical function information. Grammatical functions are annotated as edge labels that connect nodes labeled with syntactic categories. Grammatical function labels include HD (short for: head of a phrase) and nominal complement labels ON, OA, OD, and OG (short for: nominative, accusative, dative, and genitive complements). Prepositional complements can be facultative or obligatory

prepositional objects (FOPP, OPP). The most common modifiers are ambiguous modifiers (MOD) and modifiers of the verb (V-MOD); they can be nominal, prepositional, or any other syntactic category. Table 3.8 on page 18 of the TüBa-D/Z Stylebook [8] presents a complete overview of all grammatical functions that are distinguished in the TüBa-D/Z treebank.

Information about NE subclasses is appended to the syntactic category information contained in syntactic nodes. In Figure 1, NX=PER is the label of a noun phrase NX that refers to an NE of the subclass PER (short for: person), e.g. *Felix Magath*. At the part of speech level, first and family names receive the STTS tag NE (short for: proper noun). NEs can also appear in nested structures. Complex NEs can have other NEs as a proper subpart. For example, *Werder Bremen* in Figure 1 is the name of a sports club and is thus labeled as NX=ORG. This name contains the city name *Bremen*, which itself is labeled as NX=GPE.



'In [the football club] Werder Bremen, trainer Felix Magath's staff problems don't stop.'

Figure 1: Example of NE annotation in the TüBa-D/Z treebank

3 TüBa-D/Z Named Entity Annotation

NE annotation in the TüBa-D/Z has been performed manually and distinguishes between five distinct subclasses. Three of them were already illustrated in Figure 1: PER (short for: persons), such as *Felix Magath*; GPE (short for: geo-political entities) such as *Bremen*; ORG (short for: organizations, companies, etc.) such as *Werder Bremen*. In addition, the TüBa-D/Z distinguishes between LOC (short for: geographic locations) such as *Brandenburger Tor* and OTH (short for: other), the label for all NEs that cannot be subsumed under the other four NE subclasses. OTH includes inter alia book titles such as *Faust* or names of software programs such as *DOS*.

Depending on the context in which an NE is used, it may be classified as belonging to different NE subclasses. For example, if an occurrence of the NE *Forrest Gump* refers to the name of a movie, then it is classified as OTH, but when it refers to the main character in this movie, then it is labeled as PER.

The TüBa-D/Z NE annotation scheme constitutes a blend of three annotation guidelines for NE: CoNLL (www.clips.ua.ac.be/conll2003/ner/), MUC (www.cs.nyu.edu/cs/faculty/grishman/muc6.html), and ACE (www.ldc.upenn.edu/Projects/ACE). The TüBa-D/Z subclasses PER, ORG, and LOC are also found in these three guidelines. Furthermore, TüBa-D/Z follows the ACE guidelines by distinguishing between GPE (used for geographical entities with their own executive authority) and LOC (used for villages, geographical landmarks, and monuments, which don't constitute a political entity). The NE subclass OTH in the TüBa-D/Z corresponds to the CoNLL NE subclass MISC, which is not included as a separate subclass in MUC and ACE. For more information about the annotation guidelines we refer interested readers to chapter 4.2.6 of the TüBa-D/Z stylebook [8].

Table 1 provides a survey of the three NE annotated corpora currently available for German. The GerNED corpus is of rather modest size. Compared to the CoNLL-2003 dataset, TüBa-D/Z comprises almost three times as many annotated tokens. Space does not permit us to compare for languages other than German. We refer readers to the ACL anthology (http://aclweb.org/anthology-new/), the CLARIN Virtual Language Observatory (VLO; http://catalog.clarin.eu/ds/vlo/?q=named+entity), and the LDC catalogue (http://www.ldc.upenn.edu/Catalog/) for more information.

	GerNED	CoNLL 2003 data set	TüBa-D/Z, Rel. 7
PER	700	5,369	23,367
ORG	1,127	4,441	14,141
GPE	563		13,024
LOC		6,579	4,207
UKN/MISC/OTH	78	3,968	3,354
Total	2,468	20,357	58,093

Table 1: Number of instances in NE annotated corpora of German

4 Linguistic profiling of NE subclasses

State-of-the-art NE recognizers such as the Stanford Named Entity Recognizer (Finkel et al. [3]) typically utilize sequence models such as linear chain Conditional Random Fields (CRF). Such models are trained on NE-annotated data and rely on local and fairly shallow features of adjacent words: lemma information, part-of-speech, and "word-shape" properties such as the length of the word, whether the word is capitalized or not, or whether the word contains numbers of other special characters. In order to overcome

the limited size of annotated data sets and in order to be able to incorporate unannotated corpus data, semi-supervised NE recognizers such as the one by Faruqui and Padó [2] augment the sequence models with distributional generalization features obtained from large unlabeled data sets. Given this state of the art in NE recognition, it is of considerable interest to profile an NE annotated data set such as one included the TüBa-D/Z and to identify those features that are highly predictive for the classification of NEs in general and particular subclasses of NEs. The present section will summarize the empirical investigation that we conducted on the TüBa-D/Z. In subsection 4.1, we will identify several word-based co-occurrence features that corroborate the effectiveness of shallow features employed in state-of-the-art NE recognizers. In subsection 4.2, we will discuss the possibility of also utilizing deeper syntactic features that are based on the type of syntactic treebank data of the kind present in the TüBa-D/Z.

4.1 Word-based Co-occurrences

4.1.1 Appositions

In the TüBa-D/Z, NEs frequently occur in apposition constructions (marked with edge label APP), such as the one illustrated in Figure 1, where the NE *Felix Magath* of subclass PER is modified by the noun phrase NX *Trainer*. Of the 58,093 NE instances in the TüBa-D/Z, 9,260 occurrences exhibit such an appositive construction with noun phrases. It turns out that the lemmas that co-occur with NEs in such appositive constructions are disjoint and highly predictive of the five subclasses of NEs distinguished in the TüBa-D/Z (see Table 2, with numbers indicating co-occurrence frequencies).

GPE	**LOC**	**ORG**	**OTH**	**PER**
Stadt (79) 'town'	*Dorf* (20) 'village'	*Partei* (58) 'pol. party'	*Titel* (51) 'title'	*Präsident* (167) 'president'
Land (60) 'country'	*Bahnhof* (19) 'train station'	*Firma* (35) 'company'	*Motto* (45) 'slogan'	*Frau* (123) 'Ms.'
Hauptstadt (37) 'capital'	*Bezirk* (17) 'district'	*Zeitung* (32) 'newspaper'	*Thema* (29) 'topic'	*Herr* (122) 'Mr.'
Region (12) 'region'	*Stadtteil* (14) 'borough'	*Gewerkschaft* (29) 'union'	*Film* (27) 'movie'	*Sprecher* (96) 'speaker'
Republik, Staat (12) 'republic', 'state'	*S-Bahnhof* (12) 'tram stop'	*Gesellschaft* (23) 'society'	*Ausstellung* (24) 'exhibition'	*Bürgermeister* (67) 'mayor'

Table 2: NE subclasses and their most frequent nominal appositions

Geo-political entities co-occur with lemmas that refer to geographical locations, such as *Stadt* ('town'), *Land* ('country'), and *Region* ('region'), while NEs of type person tend to occur with titles such as *Präsident* ('president'), *Frau* ('Ms.'), and *Herr* ('Mr.'). This finding corroborates the effectiveness of lemma-based shallow features employed in the sequence models of current NE recognizers.

4.1.2 Verb subject combinations

Verb subject combinations show a similar co-occurrence behavior. However, unlike apposition constructions, the prototypical verbs indicative of particular subclasses of NEs are not always disjoint. For example, the verb *liegen* ('is located') typically co-occurs with both the GPE and the LOC subclasses (see Table 3). The most remarkable finding for verb subject combinations concerns the NE subclass PER. The most frequent main verb is *sagen* ('say') with 761 attested occurrences. The prediction of NE subclass PER for an NE subject with the verb *sagen* has a precision of 97.4% and a recall of 3.3% for all PER tokens. In the case of *erklären* ('explain'), precision is 89.7% with a 1.0% recall for all PER tokens. For the third most frequent verb *meinen* ('think'), precision is 95.2% and recall 0.6%.

GPE	**LOC**	**ORG**	**OTH**	**PER**
liegen (22) 'is located'	*liegen* (14) 'is located'	*berichten* (56) 'report'	*erscheinen* (18) 'appear'	*sagen* (761) 'say'
gehören (15) 'belong to'	*kommen* (7) 'come'	*machen* (44) 'make'	*zeigen* (14) 'show'	*erklären* (234) 'explain'
machen (15) 'make'	*bleiben* (6) 'stay'	*fordern* (40) 'demand'	*sehen* (12) 'see'	*meinen* (138) 'mean'
beteiligen (13) 'participate'	*mitreden* (6) 'have a say'	*lassen* (36) 'let'	*laufen* (9) 'run'	*machen* (133) 'make'
stehen (13) 'stand'	*öffnen* (6) 'open'	*spielen* (34) 'play'	*spielen*, *stehen* (9) 'play', 'stand'	*lassen* (123) 'let'

Table 3: Distributions of NE as subjects and their most frequent verbs

4.1.3 Adpositions

Adpositions (i.e. prepositions and the postpositions *zufolge* ('according to'), *gegenüber* ('towards'), and *entgegen* ('in the direction of')) are a third word class that exhibits highly predictive co-occurrences with particular subclasses of NEs. Table 4 shows three clusters of NE subclasses that are strongly associated with certain adpositions. For each cluster, there is a set of adpositions that either exhibits high precision or high recall. The adpositions

with high precision typically have a rather specific meaning and occur with low frequency in the TüBa-D/Z corpus. Adpositions with high recall, on the other hand, typically occur with high frequency in the TüBa-D/Z and express more general and thus more widely applicable meanings or are polysemous as in the case of *durch*, which can express either a locative or temporal meaning or denote the notions of instrument or agent (in the case of passive constructions).

The preposition *laut* and the postposition *zufolge* (both meaning 'according to') co-occur with 100% precision with the two NE subclasses PER and ORG. The preposition *in* ('in'), on the other hand, is highly predictive of the NE subclasses GPE and LOC with a precision of 83.8%.

Recall of NE subclasses	**Precision of adpositions**
GPE + LOC (Recall: 47.2%)	*aus* 'from' (88.6%), *in* 'in' (83.8%), *nach* 'to' (84.0%)
GPE + LOC (Recall: 0.3%)	*gen* 'toward', *entlang* 'along', *nahe* 'near', *nördlich* 'north of', *nordöstlich* 'northeast of', *nordwestlich* 'northwest of', *östlich* 'east of', *südlich* 'south of', *westlich* 'west of' (all 100%)
GPE + ORG + PER (Recall: 11.9%)	*bei* 'at'/'for' (90.7%), *für* 'for' (83.6%), *gegen* 'against' (98.1%), *mit* 'with' (84.4%), *von* 'by'/'from' (88.9%)
ORG + PER (Recall: 0,6%)	*dank* 'thanks to' (87.5%), *gegenüber* 'towards' (81.8%), *laut* 'according to' (100%), *neben* 'beside' (80.8%), *ohne* 'without' (93.3%), *seitens* 'on behalf of' (100%), *trotz* 'despite' (100%), *versus/vs.* 'versus' (100%), *zufolge* 'according to' (100%)

Table 4: Predictive prepositions for NE subclasses

4.1.4 Token Length

NEs in TüBa-D/Z consist of 1 to 15 tokens (excluding punctuation marks), with the slogan *Gegen das Vergessen ... Millionen deutsche Tote durch Vertreibung , Millionen deutsche Opfer durch den alliierten Bombenterror* ('Against oblivion… Millions of dead Germans because of displacement, millions of German victims due to the allied terror bombing') of NE subclass OTH as longest NE. Figure 2 shows that the distribution of NE subclasses varies a lot among the NEs of token length 1 to 4. While NEs of token length 2 have a strong connection with NE subclass PER, NEs of length 6 and more almost exclusively belong to the NE subclass OTH (with a precision of 84.7%). Table 5 shows the number of NEs of the different subclasses of NEs of token length 1 to 8. While there are hardly any geographical entities of NE subclasses GPE and LOC with a token length of 5, NEs of subclasses ORG and OTH consisting of five words are still relatively frequent.

These findings suggest that in addition to the "word shape" features (e.g. capitalization, cardinals) already used in state of the art NE recognizers, it is also beneficial to track the length of a candidate NE.

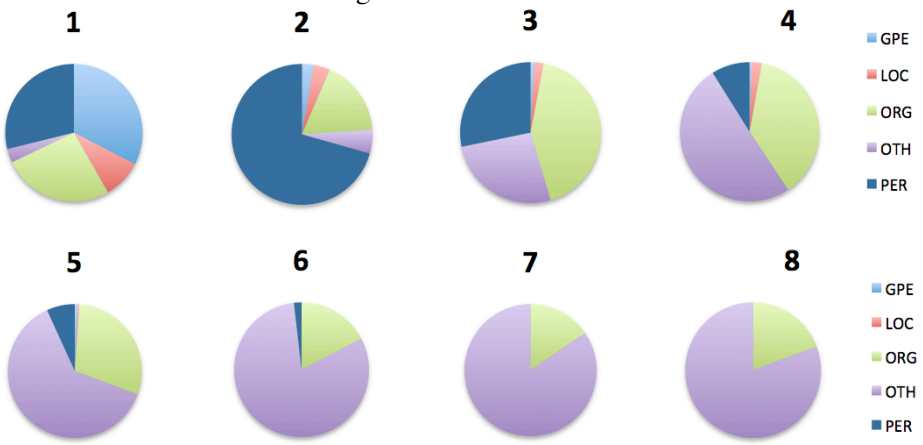


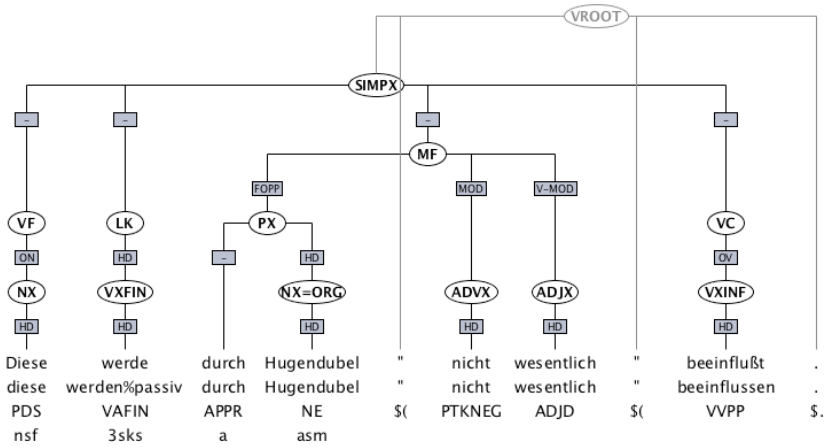
Figure 2: distribution of NE subclasses of NEs up to a length of 8 tokens

	GPE	LOC	ORG	OTH	PER
Token length 1	12,579	3,542	10,085	1,253	11,135
Token length 2	441	616	2,867	878	11,604
Token length 3	18	45	861	523	574
Token length 4	3	15	179	266	50
Token length 5	2	1	115	167	31
Token length 6	0	0	19	88	2
Token length 7	0	0	9	49	0
Token length 8	0	0	6	25	0

Table 5: Distributions of NE subclasses of NEs up to a length of 8 tokens

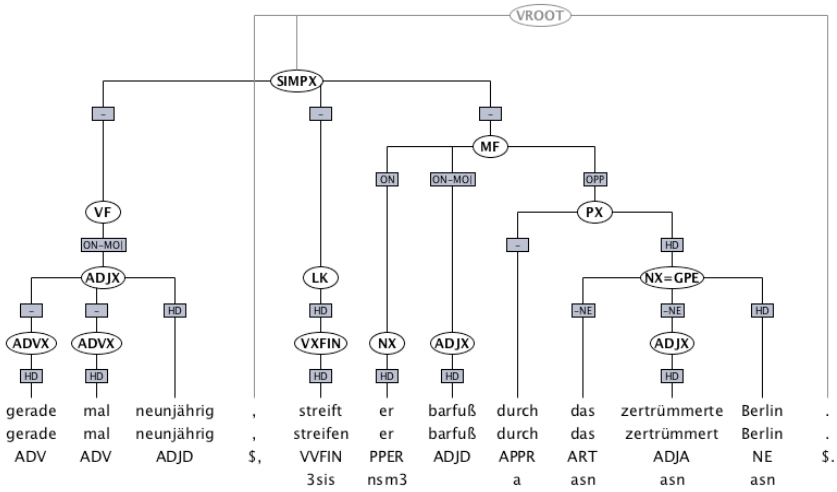
4.2 Syntax-based predictive features

In the previous subsections, we have discussed several word-based co-occurrence features for predicting subclasses of NEs. Such word-based features require, if any, only linguistic annotation at the part of speech level. However, if NE annotation is part of a treebank, as in the case of the TüBa-D/Z, it becomes possible to also consider features for predicting NE subclasses that presuppose deeper levels of annotations, such as phrase labels and grammatical functions. An empirical investigation of the syntactic environments in which NEs occur in the TüBa-D/Z has revealed that grammatical function information, in conjunction with the prepositions *durch* ('by', 'through') and *von* ('by', 'from') has a high predictive value for certain subclasses of NEs. These prepositions co-occur with NEs in three different syntactic environments: as post-head modifiers within the same noun phrase, as prepositional verbal arguments, and as verbal or sentential modifiers. These three environments are exemplified by the trees in Figures 3-5.



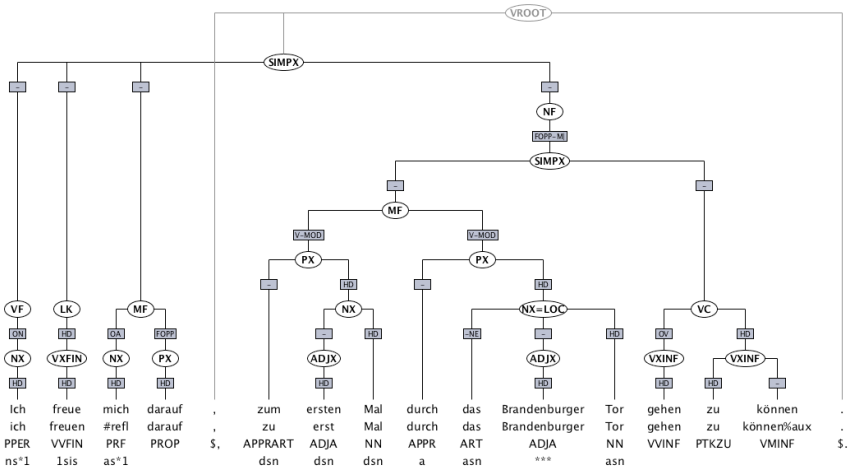
'This was "not significantly" influenced by Hugendubel.'
Figure 3: Example of grammatical function FOPP

The grammatical function FOPP is used for optional prepositional objects and for passivized subjects. The sentence in Figure 3 illustrates the use of FOPP in a prepositional phrase headed by the preposition *durch* ('by') in a passive construction.



'Having just turned nine, he strolls barefoot through the destroyed city of Berlin.'
Figure 4: Example of grammatical function OPP

The grammatical function OPP is used for obligatory prepositional objects licensed by a particular verb. In the sentence in Figure 4, the prepositional phrase headed by *durch* ('through') is licensed by the motion verb *streifen* ('stroll').



'I am looking forward to being able to walk through the Brandenburg Gate for the first time.'

Figure 5: Example of grammatical function V-MOD

The grammatical function V-MOD is used for verbal modifiers that are not obligatory. The sentence in Figure 5 illustrates this usage of *durch* ('through') as an optional modifier of the verb *gehen* ('go').

Table 6 shows that the three grammatical functions (GF) just surveyed exhibit rather distinctive distributions for the subclasses of NE differentiated in the TüBa-D/Z.

durch	FOPP	OPP	V-MOD	GF average
ORG + PER	70.8%	14.3%	29.7%	49.3%
GPE + LOC	12.5%	85.7%	62.2%	43.4%

Table 6: Distribution of NE subclasses in prepositional phrases headed by *durch*

Among the NEs that appear in prepositional phrases headed by *durch*, a large majority (70.8%) belongs to the NE subclasses of ORG and PER, when the prepositional phrase instantiates the grammatical function FOPP. This grammatical function is used for passive constructions. Therefore, the noun phrases that co-occur with *durch* in this construction have to have agent-like properties. Since such properties are typically associated with the two subclasses ORG and PER, this empirical finding is hardly surprising. On average, the subclasses of ORG and PER occur with the preposition *durch* under any of the examined grammatical functions (OPP, FOPP, PRED, V-MOD, MOD, Non-head '–') with a relative frequency of 49.3%. This contrasts sharply with the relative frequency of 70.8% for the grammatical function FOPP.

The preposition *durch* typically has the directional sense of 'through', when it appears with the grammatical functions OPP and V-MOD. Since this directional sense is consistent with noun phrases that refer to physical locations and geo-political entities, the distribution of NEs in these syntactic environments is skewed toward these two subclasses of NEs (85.7% for OPP, and 62.2% for V-MOD). Note also, that these skewed distributions vary significantly from the average of 43.4%, with which these two subclasses of NEs occur with the preposition *durch* for any of the examined grammatical functions mentioned above.

Similar facts hold for the preposition *von*, as Table 7 shows. Passivized subjects in *von*-phrases are annotated with the grammatical function FOPP; obligatory prepositional objects OPP often occur with noun phrases of NE subclasses ORG or PER. Verbal modifiers V-MOD, by contrast, contain more geographical modifiers and are more frequent than prepositional objects.

von	FOPP	OPP	V-MOD	GF average
GPE + LOC	12.8%	25.7%	38.5%	73.1%
ORG + PER	85.1%	65.7%	59.0%	49.3%

Table 7: Distribution of NE subclasses in prepositional phrases headed by *von*

5 Conclusion

This paper has provided an overview of the NE annotation included in the TüBa-D/Z treebank of German and has described the subclasses of NEs distinguished by the TüBa-D/Z annotation scheme. Two types of linguistic features have been identified that show a high correlation with particular subclasses of NE in the TüBa-D/Z. Word co-occurrence features corroborate the effectiveness of the type of surface-oriented features used in state-of-the-art sequence models for automatic NE resolution. In addition, syntax-based features have been described that go beyond the modeling currently possible with sequence models. These features demonstrate the utility of treebank annotations for the purpose of NE recognition and classification.

Future directions of research include practical experiments with the TüBa-D/Z NE data, to the best of our knowledge the largest NE annotated data set available for German, with state-of-the-art NE resolvers as well as theoretical explorations how current sequence models for NE resolution can be extended so as to be able to accommodate the type of syntax-based features made available by NE annotations in treebanks such as the TüBa-D/Z.

References

[1] Drach, Erich (1937) *Grundgedanken der Deutschen Satzlehre*. Frankfurt/Main.

[2] Faruqui, Manaal and Padó, Sebastian (2010) Training and Evaluating a German Named Entity Recognizer with Semantic Generalization. In *Proceedings of the Tenth Conference on Natural Language Processing (Konvens 2010)*, Saarbrücken, Germany, pp. 129–134.

[3] Finkel, Jenny Rose, Grenager, Trond and Manning, Christopher (2005) Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics* (ACL 2005), Ann Arbor, USA, pp. 363–370.

[4] Herling, Simon Heinrich Adolf (1821) Über die Topik der deutschen Sprache. In *Abhandlungen des frankfurterischen Gelehrtenvereins für deutsche Sprache*, pp. 296–362, 394. Frankfurt/Main. Drittes Stück.

[5] Höhle, Tilmann N. (1986) Der Begriff 'Mittelfeld'. Anmerkungen über die Theorie der topologischen Felder. In A. Schöne (Ed.) *Kontroversen alte und neue. Akten des 7. Internationalen Germanistenkongresses Göttingen*, pp. 329–340. Tübingen: Niemeyer.

[6] Ploch, Danuta, Hennig, Leonhard, Duka, Angelina, De Luca, Ernesto William and Albayrak, Sahin (2012) GerNED: A German Corpus for Named Entity Disambiguation. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC 2012)*, 20-27 May 2012, Istanbul, Turkey.

[7] Schiller, Anne, Teufel, Simone, Stöckert, Christine and Thielen, Christine (1999) *Guidelines für das Tagging deutscher Textcorpora mit STTS*. Technical report, Universities of Stuttgart and Tübingen,

Germany. URL: http://www.sfs.uni-tuebingen.de/fileadmin/static/ascl/resources/stts-1999.pdf

[8] Telljohann, Heike, Hinrichs, Erhard W., Kübler, Sandra, Zinsmeister, Heike and Beck, Kathrin (2012) *Stylebook for the Tübingen Treebank of Written German (TüBa-D/Z)*. Department of General and Computational Linguistics, University of Tübingen, Germany. URL:

http://www.sfs.uni-tuebingen.de/fileadmin/static/ascl/resources/tuebadz-stylebook-1201.pdf

[9] Tjong Kim Sang, Erik F. and De Meulder, Fien (2003) Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proceedings of the Seventh Conference on Natural Language Learning (CoNLL 2003)*, Edmonton, Canada, pp. 142–147.

Non-Projective Structures in Indian Language Treebanks

Riyaz Ahmad Bhat and Dipti Misra Sharma
Language Technology Research Center, IIIT-Hyderabad, India
E-mail: riyaz.bhat@research.iiit.ac.in,dipti@iiit.ac.in

Abstract

In recent years non-projective structures have been widely studied across different languages. These dependency structures have been reported to restrict the parsing efficiency and pose problems for grammatical formalisms. Non-projective structures are particularly frequent in morphologically rich languages like Czech and Hindi [8], [10]. In Hindi a major chunk of parse errors are due to non-projective structures [6], which motivates a thorough analysis of these structures, both at linguistic and formal levels, in Hindi and other related languages. In this work we study non-projectivity in Indian languages (ILs) which are morphologically richer with relatively free word order. We present a formal characterization and linguistic categorization of non-projective dependency structures across four Indian Language Treebanks.

1 Introduction

Non-projective structures in contrast to projective dependency structures contain a node with a discontinuous yield. These structures are common in natural languages, particularly frequent in morphologically rich languages with flexible word order like Czech, German etc. In the recent past the formal characterization of non-projective structures have been thoroughly studied, motivated by the challenges these structures pose to the dependency parsing [7], [11], [5]. Other studies have tried to provide an adequate linguistic description of non-projectivity in individual languages [4], [10]. Mannem et.al [10] have done a preliminary study on Hyderabad Dependency Treebank (HyDT) a pilot dependency treebank of Hindi containing 1865 sentences annotated with dependency structures. They have identified different construction types present in the treebank with non-projectivity. In this work we present our analysis of non-projectivity across four IL treebanks. ILs are morphologically richer, grammatical relations are expressed via morphology of words rather than the syntax. This allows words in these language to move around in the sentence structure. Such movements quite often, as we will see in subsequent sections, lead to non-projectivity in the dependency structure. We studied treebanks of four Indian Languages viz Hindi (Indo-Aryan), Urdu (Indo-Aryan), Bangla (Indo-Aryan) and Telugu (Dravidian). They all have an unmarked Subject-Object-Verb (SOV) word order, however the order can be altered under appropriate pragmatic conditions. Movement of arguments and modifiers away from the head is the major phenomenon observed that induces non-projectivity in these languages.

In this paper, we discuss the constraints and measures evaluated by [8], [12]. We evaluate these measures on IL treebanks, following with the adequate linguistic description of non-projective structures, focusing on the identification and categorization of grammatical structures that can readily undergo non-projectivity and the possible reasons for the same.

The paper is organized as follows: In Section 2, we give an overview of Indian Language Treebanking with reference to the treebanks used in this work. Section 3 discusses different constraints on dependency trees followed by the empirical results of our experiments in Section 4. In Section 5, we present an in depth analysis of non-projective structures approved by Indian Languages. Finally Section 6 concludes the paper.

2 Treebanks

In our analysis of non-projective structures in Indian languages, we use treebanks of four languages namely Hindi, Urdu, Bangla and Telugu. These treebanks are currently being developed following the annotation scheme based on the Computational Paninian Grammar (CPG) [1]. The dependency relations in these treebanks, under this framework, are marked between chunks. A chunk is a minimal, non-recursive structure consisting of a group of closely related words. Thus, in these treebanks a node in a dependency tree is represented by a chunk and not by a word. Table 1 gives an overview of the four above mentioned treebanks. While the Hindi treebanking effort has matured and grown considerably [2] the other three treebanks are still at an initial stage of development. Because of the large size and stable annotations, Hindi treebank provides major insights into potential sites of non-projectivity. In our work we have ignored intra-chunk dependencies for two reasons 1) *currently intra-chunk dependencies are not being marked in the treebanks*, and 2) *intra-chunk dependencies are projective; all the non-projective edges are distributed among the inter-chunk relations (as is the case with Hindi [10]).*

Language	Sentences	Words / Sentences	Chunks /Sentences
Hindi	20705	20.8	10.7
Urdu	3226	29.1	13.7
Bangla	1279	9.5	6.4
Telugu	1635	9.4	3.9

Table 1: IL TREEBANK STATISTICS

In comparison to other free word order languages like Czech and Danish which have non-projectivity in 23% (out of 73088 sentences) and 15% (out of 4393 sentences) respectively [8], [5], Indian languages show interesting figures, Urdu has highest number of non-projective sentences, out of 3226 sentences 23% are non-projective, in Hindi the number drops to 15% out of 20705 sentences, in Bangla 5% of 1279 sentences and interestingly there are no non-projective dependency structures in Telugu treebank.

3 Dependency Graph and its properties

In this section, we give a formal definition of dependency tree, and subsequently define different constraints on these dependency trees like projectivity, planarity and well-nestedness.

Dependency Tree : *A dependency tree D = (V, E, $\preceq$) is a directed graph with V a set of nodes, E a set of edges showing a dependency relation on V, and $\preceq$ linear order on V.* Every dependency tree satisfies two properties : a) *it is acyclic,* and b) *all nodes have in-degree 1, except root node with in-degree 0.*

3.1 Condition of Projectivity

Condition of projectivity in contrast to acyclicity and in-degree concerns the interaction between the dependency relations and the projection of a these relations on

the sequential order of nodes in a sentence.
Projectivity : *A dependency tree D = (V, E, $\preceq$) is projective if it satisfies the following condition:* $i \rightarrow j,\ \upsilon \in (i,j) \Longrightarrow \upsilon \in Subtree_i$. Otherwise D is non-projective.

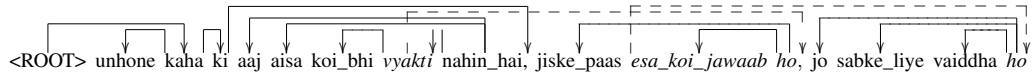
3.2 Relaxations of Projectivity

As [12] remarks, natural languages approve grammatical constructs that violate the condition of projectivity. In the following, we define the global and edge based constraints that have been proposed to relax projectivity.
Planarity : *A dependency tree D is non-planar if there are two edges* $i_1 \leftrightarrow j_1$, $i_2 \leftrightarrow j_2$ *in D such that* $i_1 \in (i_2, j_2)\ \Lambda\ i_2 \in (i_1, j_1)$. Otherwise D is planar. Planarity is a relaxation of projectivity and a strictly weaker constraint than it. Planarity can be visualized as '*crossing arcs*' in the horizontal representation of a dependency tree.
Well-nestedness : *A dependency tree is ill-nested if two non-projective subtrees (disjoint) interleave. Two disjoint subtrees* l_1, r_1 *and* l_2, r_2 *interleave if* $l_1 < l_2 < r_1 < r_2$. A dependency tree is well-nested if no two non-projective edges interleave [3].
Gap Degree : The gap degree of a node in a dependency tree is the number of gaps in its projection. *A gap is a pair of nodes* (x_n, x_{n+1}) *adjacent in* π_x *(projection of x) such that* $x_{n+1} - x_n > 1$. The gap degree of a node $gd(x_n)$ is the number of such gaps in its projection. The gap degree of a sentence is the maximum among the gap degree of its nodes [8]. Gap degree corresponds to the maximal number of times the yield of a node is interrupted. A node with gap degree > 0 is non-projective.
Edge Degree : For any edge in a dependency tree we define edge degree as the number of connected components in the span of the edge which are not dominated by the parent node of the edge. $ed_{i \leftrightarrow j}$ is the number of components in the $span(i,j)$ and which do not belong to $\pi_{parent_{i \leftrightarrow j}}$.

4 Empirical Results

In this section, we present an experimental evaluation of the dependency tree constraints mentioned in the previous section on the dependency structures across IL treebanks. Among the treebanks, Hindi treebank due to its relatively large size provides good insights into the possible construction types that approve non-projectivity in ILs. Urdu and Bangla treebanks, though comparatively smaller in size, show similar construction types approving non-projectivity. Telugu, on the other hand, as reflected by the analysis of the Telugu treebanks, does not have any non-projective structures. Possible types of potential non-projective constructions and the phenomena inducing non-projectivity are listed in Table 3. In Table 2, we report the percentage of structures that satisfy various graph properties across IL treebanks. In the treebanks, Urdu has 23%, Hindi has 15% and Bangla has 5% non-projective structures. In Hindi and Urdu treebanks, highest gap degree and edge degree for non-projective structures is 3 and 4 respectively which tallies with the previous results on Hindi treebank [10]. As shown in Table 2, planarity accounts for more data than projectivity, while almost all the structures are well-nested, Hindi has 99.7%, Urdu has 98.3% and Bangla has 99.8% of structures as well-nested. Despite the high coverage of well-nestedness constraint in these languages, there are linguistic phenomena which give rise to ill-nested structures. The almost 1% of ill-nested structures are not annotation errors but are rather linguistically justified. Few phenomena that were observed upon close inspection of the treebanks are extraposition and topicalization of verbal arguments across clausal conjunctions. Extraposition, as a reason behind ill-nestedness, is also observed by [9]. Sentence (1) shows a typical ill-nested dependency analysis of a sentence from Hindi treebank. In this sentence, **vyakti** '*person*' in complement clause is relativized by an extraposed relative clause which contains a nominal expression **esa koi jawaab** '*any such answer*' relativized by another extraposed relative clause.

(1) unhone kaha ki aaj aisa koi bhi vyakti nahin hai, jiske paas esa koi
He said that today such any EMP person not is, who near such any
jawaab ho, jo sabke liye vaiddha ho
answer has, which all for valid is
'He said that today there is no such person, who has any such answer which is valid for all.'



Languages	Properties														
	Gap Degree				Edge Degree										
	gd0	gd1	gd2	gd3	ed0	ed1	ed2	ed3	ed4	Non-proj	Non-planar	Ill-nested	Non-proj & Planar	Non-proj Edges	Total Sentences
Hindi	85.14	14.56	0.28	0.02	85.14	14.24	0.45	0.11	0.03	14.85	13.62	0.19	1.24	1.65	20497
Urdu	77.85	20.58	1.31	0.12	77.85	19.20	1.97	0.56	0.22	22.12	20.11	1.66	2.00	2.59	3192
Bangla	94.45	5.47	0.0	0.0	94.45	5.24	0.16	0.08	0.0	5.47	3.91	0.16	1.25	0.97	1279
Telugu	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1606

Table 2: Non-projectivity measures of Dependency Structures in IL Treebanks

Construction Type	Phenomenon																	
	Topicalisation			Extraposition			NP Extraction			Quantifier Float			Scrambling			Inherent		
	Hindi	Urdu	Bangla	Hindi	Urdu	Bangla	Hindi	Urdu	Bangla	Hindi	Urdu	Bangla	Hindi	Urdu	Bangla	Hindi	Urdu	Bangla
Genitive Constructions	-	-	-	-	1	-	327	232	23	-	-	-	-	-	-	-	-	-
Relative Clauses	-	-	-	999	240	15	-	-	-	-	-	-	-	-	-	-	-	-
Conditionals	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	496	252	13
Clausal Complements	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1555	361	23
Control Constructions	12	-	-	-	-	-	-	-	-	-	-	-	39	-	-	-	-	-
Co-ordination	-	-	-	-	-	-	-	-	-	-	-	-	10	-	-	-	-	-
Quantified Expressions	-	-	-	-	-	-	-	-	-	12	4	-	-	-	-	-	-	-
Other Finite Clauses	88	54	13	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Others	-	-	-	-	-	-	-	-	-	-	-	-	2	1	5	-	-	-

Table 3: Categorization of Construction Types and Phenomena behind Non-projectivity in IL treebanks.

5 Analysis and Categorization

In this section we discuss different types of constructions that allow non-projectivity and the linguistic phenomena that induce non-projectivity in them. Our study of IL treebanks revealed a number of construction types with non-projectivity namely *Genitive Constructions, Relative clause constructions, Conditionals, Clausal complements, Control Constructions, Co-ordinated constructions, Quantified expressions and, Other Finite Clauses.* Some of these formatives are inherently discontinuous like conditionals, however a majority of them, with canonical order projective, can be rendered non-projective under appropriate pragmatic conditions via movement. A number of movement phenomena observed behind non-projectivity in IL treebanks are: *a) Topicalisation, b) Extraposition, c) Quantifier floating, d) NP Extraction, e) Scrambling- any movement other than 'a – d'.*

Below we discuss a few of the above mentioned construction types and the reasons behind the non-projectivity in them. The examples discussed are from Hindi and Urdu Treebanks.

5.1 Relative Clause Constructions

In Hindi, Urdu and Bangla relative clauses have three different orders, they can be left adjoined-placed immediately before their head noun; embedded-placed immediately after the head noun and extraposed-placed post-verbally away from the head noun. Since extraposed relative clauses are separated from the head noun, this dislocation generates discontinuity in the structure. In example (2), the nominal expression **mojuda swaroop** '*current form*' in the main clause is modified by the extraposed relative clause. The projection of the head noun **mojuda swaroop** '*current form*' is interrupted by its parent **hai** '*is*'. Although it is mainly an

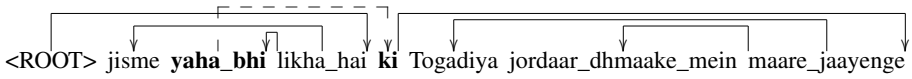
extra-posed relative clause that generates discontinuity, there are instances in the IL treebanks where even Left-adjoined relative clauses can be separated from their head noun by some verbal argument.

(2) iskaa mojuda swaroop theory aadharit hai jisko practical aadhirit banaaya jaayega
Its current form theory based is which practical based made will be
'Its current form is theory based which will be made practical based.'

5.2 Clausal Complements

Clausal complements, introduced by a complementizer (ki in Hindi/Urdu, je in Bangla), are placed post-verbally in Hindi-Urdu and Bangla. If the head predicate licensing the clausal complement is other than the verb, the canonical order is such that the head is positioned preverbally and its complement is extraposed. In such order the structure has inherent discontinuity. Example (3) shows an extraposed complement clause of an expletive **yaha** '*it*'. The head element and the complement clause are at a distance from each other, the verb **likha hai** '*is written*' in the main clause interferes in the projection of **yaha** '*it*' making the structure discontinuous. Extraposed complement clauses are the major source of non-projective structures in IL treebanks. In Hindi treebank around 42% non-projective structures are due to extraposed clausal complements of a non-verbal predicate.

(3) jisme yaha bhi likha hai ki Togadiya jordaar dhmaake mein maare jaayenge
In which this also written is that Togadiya powerful blast in killed will be
'In which this is also written that Togadiya will be killed in powerful blast.'



5.3 Genitive Constructions

In genitive constructions, the genitive marked nominal is easily dislocated from the head noun. The study of IL treebanks show a varied number of movements from genitive constructions. Genitive marked noun can either be extraposed or be extracted towards the left. However, the extraction towards left is wide spread with good number of instances in all the treebanks except Telugu treebank. In example (5), genitive marked pronoun **jiski** '*whose*' has been extracted from its base position to the sentence initial position crossing the subject of the sentence.

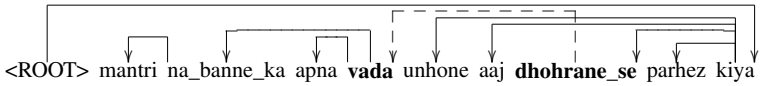
(4) jiski raashtra ko bhaari keemat adaa karni padi
for which country ACC heavy cost pay had to
'For which country had to pay a heavy cost'

5.4 Control Constructions

In ILs under study, verbs can select non-finite complements and adverbial clauses marked with infinitive or participle inflections *(-kar and -na in Hindi-Urdu)*. In such bi-clausal combinations non-finite clauses have a null subject controlled by a syntactic argument of the main verb. In IL treebanks such arguments, which thematically belong to both the verbs but are syntactically governed only by the main verb, are annotated as the child of the main verb only in view of the single-headedness constraint of dependency trees. Interestingly, in these control constructions, individual arguments of non-finite verb can move around and cross the shared argument, child of the main verb, generating discontinuity in non-finite clause. There are varied occurrences of such discontinuous non-finite clauses in

IL treebanks. Example (6) is a reflection of such discontinuity in these treebanks. In this example, the complement of the verb **dhorana** '*to repeat*' has moved past the shared arguments, **unhone** '*he*' and **aaj** '*today*', between the non-finite verb **'dhorana'** '*to repeat*' and the main verb **'kiya'** '*did*' leading to discontinuity in the non-finite clause.

(5) mantri na banne ka apna vada unhone aaj dhorane se perhez kiya
minister not become of his promise he ERG today repeat ABL refrain did
'Today he refrained from repeating his promise of not becoming the minister.'



Conclusion

In this paper, we have looked into the constraints and formal properties of dependency structures across Indian languages that have been defined in the literature on dependency grammars. We did an in depth study of dependency structures, that allow non-projectivity, identifying the possible reasons that these languages offer for discontinuous yield of governing categories. We have identified a list of grammatical formatives that are potential sites of discontinuity in the closely related languages namely Hindi, Urdu and Bangla. Important goal for future will be to incorporate the analysis done in this paper into the dependency parsing of these languages.

Acknowledgments

The work reported in this paper is supported by the NSF grant (Award Number: CNS 0751202; CFDA Number: 47.070). [1]

References

[1] R. Begum, S. Husain, A. Dhwaj, D.M. Sharma, L. Bai, and R. Sangal. Dependency annotation scheme for indian languages. In *Proceedings of IJCNLP*, 2008.

[2] R. Bhatt, B. Narasimhan, M. Palmer, O. Rambow, D.M. Sharma, and F. Xia. 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, 2009.

[3] M. Bodirsky, M. Kuhlmann, and M. Möhl. Well-nested drawings as models of syntactic structure. In *Tenth Conference on Formal Grammar and Ninth Meeting on Mathematics of Language*, pages 195–203, 2009.

[4] E. Hajicová, J. Havelka, P. Sgall, K. Veselá, and D. Zeman. Issues of projectivity in the prague dependency treebank. *Prague Bulletin of Mathematical Linguistics*, 81:5–22, 2004.

[5] J. Havelka. Beyond projectivity: Multilingual evaluation of constraints and measures on non-projective structures. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, volume 45, page 608, 2007.

[6] S. Husain and B. Agrawal. Analyzing parser errors to improve parsing accuracy and to inform tree banking decisions. *Linguistic Issues in Language Technology*, 7(1), 2012.

[7] M. Kuhlmann and M. Mohl. Mildly context-sensitive dependency languages. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, volume 45, page 160, 2007.

[8] M. Kuhlmann and J. Nivre. Mildly non-projective dependency structures. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 507–514. Association for Computational Linguistics, 2006.

[9] W. Maier and T. Lichte. Characterizing discontinuity in constituent treebanks. In *Formal Grammar*, pages 167–182. Springer, 2011.

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

[11] R. McDonald and J. Nivre. Characterizing the errors of data-driven dependency parsing models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 122–131, 2007.

[12] J. Nivre. Constraints on non-projective dependency parsing. In *Eleventh Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 73–80, 2006.

[1] 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.

Experiments on Dependency Parsing of Urdu

Riyaz Ahmad Bhat, Sambhav Jain and Dipti Misra Sharma
Language Technology Research Center, IIIT-Hyderabad, India
{riyaz.bhat|sambhav.jain}@research.iiit.ac.in, dipti@iiit.ac.in

Abstract

In this paper, we present our parsing efforts for Urdu, a South Asian language with rich morphology. In this effort we tried to identify the important aspects of the morphology in Urdu which could be exploited for efficient syntactic parsing. The most important feature is the presence of case clitics on nominals which mark the kind of relation they bear to their heads. In Urdu, unlike a positioned language, arguments of the verb are expressed not through their absolute position in the sentence but through the morphology they carry. Experiments using the Urdu dependency treebank (UDT) show the significant impact of the case markers on parsing accuracy of Urdu.

In this paper we have experimented with dependency parsing of Urdu using the Urdu Dependency Tree-bank (UDT). In UDT there are 3226 sentences (approx. 0.1M words) annotated at multiple levels viz morphological, part-of-speech (POS), chunk and dependency levels. Apart from parsing experiments we also reported some of the problem areas and issues concerning the dependency parsing of Urdu.

1 Introduction

Parsing morphologically rich languages (MRLs) like Arabic, Czech, Turkish, etc., is a hard and challenging task [9]. A large inventory of word-forms, higher degrees of argument scrambling, discontinuous constituents, long distance dependencies and case syncretism are some of the challenges which any statistical parser has to met for efficient parsing of MRLs. Due to the flexible word order of MRLs, dependency representations are preferred over constituency for their syntactic analysis. The dependency representations do not constraint the order of words in a sentence and are thus better suited for the flexible ordering of words in such languages. Like any other MRL, Indian languages are rich in morphology and allow higher degrees of argument scrambling. [1] have proposed a dependency based annotation scheme for the syntactic analysis of Indian languages. Currently a number of dependency tree-banks are under development following the annotation scheme. Urdu treebank is one among the tree-banks under development which we have used in this work [3].

In recent times the availability of large scale syntactic tree-banks has led to a manifold increase in the development of data driven parsers. CoNLL shared task

2006 and 2007 have addressed the task of data driven dependency parsing for two consecutive years. Among other shared tasks are EVALITA 2007 and 2009 Dependency Parsing Task, ICON 2009 and 2010 tool contest for Indian Language Dependency Parsing. Broadly two different algorithmic approaches, i.e. Graph Based [8] and Transition Based [11]) have been employed for the data driven dependency parsing. We have chosen state of the art parser from each of the above approach, namely MaltParser ([10] and MSTParser [8]) for experiments reported in this paper.

In the following section we present some of the challenges that Urdu poses for parsing. Section 3 gives an overview of the experimental setup followed by discussion in Section 4. Section 5 will finally conclude the paper.

2 Challenges in Parsing Urdu

As mentioned earlier there are a number of challenges for parsing morphologically rich languages. Urdu due to its morphological rich nature poses all such problems for its syntactic parsing. Some of the problems are discussed in this section:

Non-projectivity: Non-projective structures in contrast to projective dependency structures contain a node with a discontinuous yield. The phenomenon of non-projectivity poses problems for both grammar formalisms as well as syntactic parsing [5]. Among the 3226 sentences in UDT, 22% of the sentences have non-projective structures and 2.5% structures are non-projective.

Argument Scrambling: In Urdu because case markers carry the information about the relation between words, these words can freely change their positions in the sentence. Argument scrambling often leads to discontinuities in syntactic constituents and many a times leads to non-projective structures and long distance dependencies thus posing a challenge to parsing. Given appropriate pragmatic conditions a simple sentence in Urdu allows n [1] factorial ($n!$) permutations.

Case Syncretism: In Urdu case markers and case roles do not have a one to one mapping, each case marker is distributed over a number of case roles. Among the six case markers only Ergative case marker is unambiguous [3]. Although case markers are good indicators of the relation a nominal bears in a sentence, the phenomenon of case syncretism bars their ability in effectively identifying the role of the nominal while parsing.

Data Sparsity: In Urdu words take various lexical forms based on the grammatical information they carry. In case of nouns, they decline to reflect number and case, while verbs inflect for TAM (tense, aspect and modality) and carry agreement features (gender, number and person) of one of its arguments. Such multiple forms of words increase the vocabulary size of the language, which causes the data sparsity problems in a number of natural language processing applications including parsing. In Urdu the coverage of a previously unseen text is very less to the usual coverage of an analytical language. In this work we used the standard mechanisms

[1] n is the number of chunks in the sentence.

for mitigating the effects of sparse data on parsing like POS tag information and word similarity (by including lemma information).

3 Experiments

In this section we will discuss in detail the basis for corpus split and the feature selection and discuss the motivation and impact of such split and selection on the issues raised in the previous section.

3.1 Data Split

Annotations in UDT are represented in SSF format, for experimentation UDT was converted to CoNLL-X format and split into training, testing and development sets. The training set has 2258 sentences, that is approximately 70% of the total corpus. The development and training sets have each of 484 sentences i.e. each set is around 15% of the total corpus. Training-test split was not random, we used the stratified partitioning technique[2] to select the sentences. This ensures that each of the three sets have approximately the same distribution of the parameters (considered for stratification) and thus we have a better representative training and testing data sets. The parameters which we choose for stratification are: **Sentence Length** and **Non Projectivity**. Average sentence length (in terms of chunks/nodes) in UDT is 13.8 with standard deviation of 7.6, stratified sampling ensured training-test split with a distribution of sentence of almost similar average length and same standard deviation across training, testing and development sets. Likewise non-projective sentences were distributed with 70%, 15% and 15% approximately among the training, testing and development sets out of 694 sentences.

3.2 Feature Selection

Through experimentation we have tried to analyze the impact of both the linguistic information as well as the algorithmic and parser settings. Under linguistic information we have studied the efficacy of POS tag information, word's lemma information, morphological features in the form of case markers, tense, aspect and modality (TAM) and general gender-number-person (GNP) information. Since each node of a tree represents a chunk in UDT [3], word that projects the chunk i.e. the head of the chunk[3], is treated as a node in CoNLL-X format, removing all other words (adjectives, quantifiers, intensifiers)[4] in the chunk except case and TAM words in order to exploit their role in label and head prediction while parsing. Case markers and TAM are copied on to the head word of the respective chunk as a feature. POS tags, chunk tags (CPOS), lemma and GNP features used in this

[2]Stratified Sample is a sample that is selected to ensure that certain categories and cases are included, proportionate to their presence in the population.

[3]Chunk heads are determined automatically from the gold chunked data.

[4]Local words in the chunk do not contribute to the global dependency of the chunk in the sentence.

work are of gold standard. The effect of each feature is presented in a consolidated results Table 1.

3.3 Parser Settings

We have used MaltParser version 1.6.1 and MSTParser version 0.5.0 for experiments reported in this paper. For some initial experiments we used “nivreeager" and “nivrestandard" parsing algorithm, however we observed that the former consistently outperformed the latter. The observation is consistent for most MRL's [2]. Hence, for all our experiments with MaltParser we used “nivreeager" parsing algorithm with default SVM setting parameters. Feature model in MaltParser acts as the template for learning. We have adopted the feature model used by [4] for Hindi since it is the state of the art parser for Hindi, and Urdu is typologically related to Hindi [6, p. 27]. In MSTParser we used the *k=5*, it has been reported to fetch best results ([7], [2]). The default order=1 is used. To handle non-projective structures in Urdu, we used non-projective settings of both the parsers for all the experiments except for the baseline[5].

We have laid out the experiments in a way that each experiment conveys the extent of usefulness and information gain using a feature or a setting from the perspective of dependency parsing. For each experiment, the settings and features used in the previous experiment were retained.

	MaltParser			**MSTParser**		
	LAS (%)	UAS (%)	LAcc (%)	LAS (%)	UAS (%)	LAcc (%)
Baseline	58.05	75.34	61.34	51.39	71.92	55.72
Non-Projective Settings	58.89	76.39	62.24	51.84	72.43	55.92
Optimized Parser Settings	59.72	75.18	63.51	51.76	72.75	56.01
POS-tags	64.99	81.12	67.50	62.55	80.16	65.32
Lemma	65.87	81.24	68.38	62.91	62.91	65.61
Case and TAM Markers	**76.61**	**88.45**	**80.03**	**67.29**	**87.30**	**70.07**
GNP	76.15	88.11	79.54	66.72	86.53	69.32

Table 1: Results of all the experiments with MaltParser and MSTParser

4 Results

Table 1 shows the accuracy of both the MaltParser and the MSTParser. The MaltParser has outperformed the MSTParser in all the experiments. The best result

[5]To set the baseline we used the default settings of the parsers. The data has only the surface lexical form (FORM) of words.

obtained with MALTParser is 76.61% label attachment score, 88.45% unlabeled attachment score, 80.03% label accuracy score. The best result for both the parsers is obtained on the same experimental settings incorporating case information, which further strengthens our conjecture on the importance of case markers. All the experiments suggest the effectiveness of the features successively included in the system. By incorporating the POS-tag, chunk tag, lemma and case markers both the parsers have shown a steep increase in their performance. However, GNP information had a negative impact on the performance of both the parsers. In Urdu verb mainly agrees with its nominative argument which can either be an 'agent' or a 'theme'. In sentences where both the arguments of a verb (transitive) are in nominative form, it agrees with 'agent'. A verb can also agree with a constituent inside its clausal complement, the phenomenon called Long Distance Agreement. It's probably because of this agreement behavior of a verb which affects the parsing accuracy. Among all the features incorporated, case markers have played a major role in improving the parsing performance. A 10.74% increment in LAS in MaltParser with case information makes a clear statement about the importance of morphology based parsing of MRLs.

During the error analysis, the primary confusion is observed due to granularity of the tag-set, for labels 'r6' and 'r6-k2' (genitive relations) for example a total of 50 cases have been interchangeably marked incorrect. This can be explained by the fact that the labels differ slightly in their semantics and it is potentially not possible to disambiguate based on the simple features used in the experiments. The issue of granularity will automatically subside if a coarse grained tag-set is used ignoring finer distinctions.

5 Conclusion and Future Work

This paper presents our efforts towards the dependency parsing of Urdu. It is an attempt to explore the importance of linguistic information encoded in the morphology of the language for data driven parsing. Our main inference from the experiments is that some morphological features viz. the case markers play a vital role in parsing while on the other hand morphological information of gender, number and person (agreement features), has not delivered any improvement. In the process we have also investigated the extent of impact of non-projectivity, and inspected the role of POS tags and lemma on parsing accuracy.

We have currently reported our work on the chunk heads, which we wish to extend to full parsing with chunks expanded along with their intra-chunk dependencies. Also the current dependency tag-set is highly fine grained, consisting of 53 tags, and often for practical applications we do not need such deep analysis. So, our efforts would be to come up with an efficient parser with coarse grained dependency labels.

Acknowledgments

The work reported in this paper is supported by the NSF grant (Award Number: CNS 0751202; CFDA Number: 47.070). [6]

References

[1] R. Begum, S. Husain, A. Dhwaj, D.M. Sharma, L. Bai, and R. Sangal. Dependency annotation scheme for indian languages. In *Proceedings of IJCNLP*. Citeseer, 2008.

[2] A. Bharati, S. Husain, B. Ambati, S. Jain, D. Sharma, and R. Sangal. Two semantic features make all the difference in parsing accuracy. *Proc. of ICON*, 8, 2008.

[3] R.A. Bhat and D.M. Sharma. A dependency treebank of urdu and its evaluation. *In Proceedings of 6th Linguistic Annotation Workshop (ACL HLT 2012). Jeju, Republic of Korea. 2012.*

[4] P. Kosaraju, S.R. Kesidi, V.B.R. Ainavolu, and P. Kukkadapu. Experiments on indian language dependency parsing. *Proceedings of the ICON10 NLP Tools Contest: Indian Language Dependency Parsing*, 2010.

[5] M. Kuhlmann and M. Mohl. Mildly context-sensitive dependency languages. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, volume 45, page 160, 2007.

[6] C.P. Masica. *The Indo-Aryan Languages*. Cambridge Univ Pr, May 1993.

[7] R. McDonald, K. Crammer, and F. Pereira. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 91–98. Association for Computational Linguistics, 2005.

[8] R. McDonald, F. Pereira, K. Ribarov, and J. Hajič. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 523–530. Association for Computational Linguistics, 2005.

[9] J. Nilsson, S. Riedel, and D. Yuret. The conll 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL*, pages 915–932, 2007.

[10] J. Nivre, J. Hall, J. Nilsson, A. Chanev, G. Eryigit, S. Kübler, S. Marinov, and E. Marsi. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(02):95–135, 2007.

[11] J. Nivre, J. Hall, J. Nilsson, G. Eryiğit, and S. Marinov. Labeled pseudo-projective dependency parsing with support vector machines. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, pages 221–225. Association for Computational Linguistics, 2006.

[6] 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.

<tiger2/> as a standardised serialisation for ISO 24615 – SynAF

Sonja Bosch, University of South Africa (UNISA) in Pretoria – Key-Sun Choi, KAIST – Éric de la Clergerie, Inria – Alex Chengyu Fang, City University of Hong Kong – Gertrud Faaß, University of Hildersheim – Kiyong Lee, Korea University – Antonio Pareja-Lora, Universidad Complutense de Madrid – Laurent Romary, Inria & Humboldt University – Andreas Witt, Institut für Deutsche Sprache – Amir Zeldes, Humboldt Universität zu Berlin – Florian Zipser, Inria & Humboldt Universität zu Berlin

Abstract

This paper presents the application of the <tiger2/> format to various linguistic scenarios with the aim of making it the standard serialisation for the ISO 24615 [1] (SynAF) standard. After outlining the main characteristics of both the SynAF metamodel and the <tiger2/> format, as extended from the initial Tiger XML format [2], we show through a range of different language families how <tiger2/> covers a variety of constituency and dependency based analyses.

1 From SynAF to <tiger2/>

In 2010, the International Organization for Standardization (ISO) published ISO 24615 (SynAF), which provides a reference model for the representation of syntactic annotations. This document elaborates on early proposals for the representation of syntactic information [3] and [4] to provide a comprehensive framework based on the following principles:

- Dealing with the representation of both syntactically annotated corpora and data resulting from an automatic syntactic analysis
- Equally covering dependency and constituency based representations
- Being flexible enough to deal with a variety of languages by relying on an open metamodel which can be parameterized by means of specific data categories

Still, the SynAF standard did not contain any specific XML serialisation. This prevented it from being sufficient for implementers who would want to ensure that their data be interoperable with other initiatives, at least for the same language. This is why ISO subcommittee TC 37/SC 4 initiated a new activity, in order to provide an additional part to SynAF on the basis of the Tiger XML format [2]. This format has been slightly adapted[1] in the

[1] All background information on <tiger2/> can be found at http://korpling.german.hu-berlin.de/tiger2/, comprising schemas, documentation and encoded examples.

meantime under the code name <tiger2/>, to comply with recent XML technology developments[2], to integrate some necessary features and to fully encompass dependency-based representations.
Before proceeding any further with this initiative, the SC4 experts who attended the ISO TC37 meeting in Madrid in June 2012 decided to put <tiger2/> to the test, by applying it to a variety of language samples. This should provide a concrete basis for the future committee work, which should integrate the feedback from the community interested in syntactic annotation and in particular colleagues involved in the development of treebanks. In this context, this paper is intended to provide a first overview of these experiments, with the aim of raising some interest and thereby getting some feedback from a variety of interested experts.
This paper will first of all describe the major characteristics of both the ISO SynAF standard and the <tiger2/> formats and then present a series of short sections demonstrating a) the possible application of <tiger2/> in specific linguistic contexts; and b) how SynAF can be used as an interchange format for existing tools. As mentioned, this paper discusses work in progress. Consequently, we expect it to lead other colleagues to start experimenting with <tiger2/> and, thus, provide some concrete feedback to the forthcoming standardisation work.

2 The SynAF metamodel

ISO 24615 (SynAF) provides a high-level metamodel for representing the syntactic annotation of linguistic data, with the objective of supporting interoperability across language resources or language processing components. SynAF aims at covering the two main functions of syntactic annotation in language processing, namely: a) to represent linguistic constituency, as in Noun Phrases (NP) etc., describing a structured sequence of morpho-syntactically annotated items (including, depending on the theory used, empty elements or traces generated by movements at the constituency level), as well as constituents built from non-contiguous elements; and b) to represent dependency relations, such as head-modifier relations, and also including relations between categories of the same kind (such as the head-head relations between nouns in appositions, or nominal coordinations in some formalisms).
As a consequence, syntactic annotations must comply with a multi-layered annotation strategy, which interrelates syntactic annotation for both

[2] Typically: generic XML attributes, schema technology, XML design good practices. Although it is not the aim of this paper to go in to the corresponding technical details, we encourage interested readers to refer to the <Tiger2/> documentation

constituency and dependency, possibly simultaneously, as stated in the SynAF metamodel. The metamodel is based on the following components:

- The *T_node* component represents the terminal nodes of a syntactic tree, consisting of morpho-syntactically annotated word forms, as well as empty elements when appropriate. T_nodes are annotated with syntactic categories valid for the word level.
- The *NT_node* component represents the non-terminal nodes of a syntactic tree. The NT_nodes are annotated with syntactic categories that are valid at the phrasal, clausal and/or sentential levels.
- The *Edge* component represents a relation between syntactic nodes (both terminal and non-terminal nodes). For example, the dependency relation is binary, consisting of a pair of source and target nodes, with one or more annotations.

From this metamodel, a specific syntactic annotation model can be obtained by combining the above-mentioned components with data categories characterising or refining their semantics.
It should be noticed that the terminal node level in SynAF is strongly equivalent to the word form level in MAF (ISO 24611 [5]), for which we offer concrete interfaces below. It is thus left to the implementer to either separate or merge these two components depending on whether it is relevant or not, for instance, to clearly differentiate the data categories attached to word forms and terminals within a multi-layered annotated corpus.

3 <tiger2/> as a SynAF compliant Tiger

The main characteristics of the <tiger2/> datamodel can be summarized as follows:

- Terminal nodes are implemented as <t> elements, either referring to a textual segment or pointing to a word form in a morphosyntactically annotated corpus. Non-terminal nodes are implemented as <nt> elements and are used to represent hierarchical structures like syntactic trees
- The Edge component is implemented by means of an <edge> element, which may relate either <nt> or <t> elements to each other.
- A generic @type attribute inside terminal, non-terminal and edge elements can further qualify the meaning of these elements. For instance a non-terminal node can be qualified as a verbal complex, a terminal can be specified as a compound node etc.
- The elements terminal, non-terminal and edge as well are used as placeholders for further linguistic annotations. Such annotations can freely be added as generic attribute-value-pairs with no restrictions to

a specific linguistic school or theory (e.g. attributes like @cat, @phrase, … or any user defined attributes).

- In addition to corpus metadata, an annotation block at the beginning of the document allows to declare the particular descriptive features used in the annotation project.
- An important change from the initial Tiger format [2] is that with <tiger2/> it is possible to directly implement edges between terminal modes, thus empowering the datamodel with a fully-fledged dependency-based representation. As demonstrated in section 4.2 for Semitic languages, such edges may appear as children of <t> elements. They may also be used for further sentential and even inter-sentential relations such as binding and anaphora (see also subsections 4.1.4 and 4.3.5).

All these features have been implemented on the basis of a complete specification written in the TEI ODD language from which one can generate both various schemas (DTD, RelaxNG, W3C) and the actual documentation. Since not all the features integrated in <Tiger2/> may be needed at the same time for a specific implementation, it is anticipated that project specific customizations, based on the ODD developed made so far, will allow the community to converge on specific flavours of <Tiger2/>.

4 <tiger2/> at work – linguistic cases

At this stage, we do not want to discuss the actual coverage and expressive power of the <tiger2/> proposal further without validating it across a variety of possible languages. To this end, the authors of this paper have volunteered to test the <tiger2/> proposal on the languages where they had the appropriate expertise. This is reflected in this section, where a quick overview of the main characteristics and challenges is given for a sample of European, African and Asian languages, followed by the explicit representation of an example in <tiger2/>. Even if the actual validation of the model will only occur when wide-coverage syntactic analyses or treebanks will be transformed into this format[3], it will allow us to estimate in which direction the work on the future second part of SynAF should go.

4.1 Spanish and Romance Languages

The set of syntactic phenomena that are present in Romance languages do not differ much from those present in English. Hence, their representation does not require adding many other particular devices to the ones included in

[3] and accordingly a comprehensive set of data categories for such languages will be made available in ISOCat

SynAF and/or in <tiger2/>. However, they share some common particularities that will be discussed next, namely enclitic objects and contractions, elliptic subjects and redundancy, which can be described using a combination of MAF and SynAF [6], [7] and [8]. Each of them is presented in a dedicated subsection below.

4.1.1 Spanish enclitic objects

First, most Romance languages allow clitic constructions, much like Semitic languages (see Section 4.2). For example, the Spanish sentence 'Díselo al médico' (= 'Tell [to him] it to the doctor) includes a couple of enclitic pronouns ('se' and 'lo') attached to the word form 'Dí', which is the main verb of the sentence. This phenomenon can be found also in Italian and in Catalan, amongst others. The morphosyntactic segmentation of this sentence is shown in Example 1, using the encoding scheme of MAF, which also shows the particular morphosyntactic annotation of 'Díselo'.

```
<maf>
  <token xml:id="t1">Dí</token>      <!-- Dí = Tell   -->
  <token xml:id="t2">se</token>      <!-- se = to him -->
  <token xml:id="t3">lo</token>      <!-- lo = it   -->
  <token xml:id="t4">a</token>       <!-- a = to    -->
  <token xml:id="t5">l</token>       <!-- el = the  -->
  <token xml:id="t6">médico</token>  <!-- médico = doctor -->
  <token xml:id="t7">.</token>

  <!-- wordForm for "Díselo", which is the aggregation of -->
  <!-- the verbal form "Dí" and the enclitic pronouns -->
  <!-- "se" and "lo".                                 -->
  <wordForm xml:id="wordForm1" tokens="t1 t2 t3"/>
    <!-- wordForms for "Dí", "se" and "lo" -->
    <wordForm xml:id="wordForm2" lemma="decir" tokens="t1">
      <fs>
        <f name="pos">
          <symbol value="VMI2S"/>
          <!-- Verb-Main,Imperative,2nd person,Singular -->
        </f>
      </fs>
    </wordForm>
    <wordForm xml:id="wordForm3" lemma="se" tokens="t2">
      <fs>
        <f name="pos">
          <symbol value="PP3S"/>
          <!-- Personal Pronoun, 3rd person, Singular -->
         </f>
      </fs>
    </wordForm>
    <wordForm xml:id="wordForm4" lemma="lo" tokens="t3">
      <fs>
        <f name="pos">
          <symbol value="PP3S"/>
          <!-- Personal Pronoun, 3rd person, Singular -->
        </f>
      </fs>
    </wordForm>
  </wordForm>
  <!-...->
```

```
</maf>
```

Example 1: Morphosyntactic segmentation of the Spanish sentence 'Díselo al médico' and morpho-syntactic annotation of the clitic construction 'Díselo'

4.1.2 Contractions

Second, most Romance languages include some contractions, that is, some word forms consisting of the fusion of a preposition and an article. This is not particular to Romance languages, since English and German (for example) include this type of words too. Example 2 shows how the Spanish contraction 'al' (= 'to the', which corresponds also to 'au' in French, for instance) has been annotated conforming to MAF.

```
<maf>
  …
  <!-- wordForm for "al", which is the contraction of  -->
  <!—the preposition "a" and the definite article "el" -->
  <wordForm xml:id="wordForm5" tokens="t4 t5"/>

    <!-- wordForms for "a" and "el" -->
    <wordForm xml:id="wordForm6" lemma="a" tokens="t4"/>
      <fs>
        <f name="pos">
          <symbol value="AP"/>      <!-- Adposition (Preposition) -->
        </f>
      </fs>
    <wordForm xml:id="wordForm7" lemma="el" tokens="t5"/>
      <fs>
        <f name="pos">
          <symbol value="ATDMS"/>
          <!-- Article-Definite, Masculine, Singular -->
      </f>
      </fs>
  </wordForm>
  …
</maf>
```

Example 2: Morphosyntactic annotation of the Spanish contraction 'al'

4.1.3 Elliptic subjects

Third, Romance languages are characterized by the agreement in person and in number between the subject and the main verb of the sentence, and some of them (like Spanish) also require the subject and the main verb to agree in gender in certain cases. Besides, the inflection of Spanish verbs usually identifies unambiguously the person and the number of the subject. Due to this, Spanish subjects are elliptic in most cases. In fact, if the subject is made explicit in such cases, it can be considered redundant and, hence, pragmatically emphasized. Therefore, a mechanism for making elliptic subjects explicit might be required in order to annotate the full structure and/or set of dependencies of Spanish sentences. This mechanism would be similar to the one shown in Example 1 and 2 for Buntu (see 4.3.5), which

shows how some elements that are not present on the surface structure can be made explicit by means of the standard.

4.1.4 Grammar-required redundancy

Fourth, the enclitic constructions mentioned in Section 4.1.1 usually entail the inclusion of a redundant syntactic element, like the pronoun ‘se’ in the Spanish sentence ‘Díselo al médico’. This a personal pronoun, which coreferences the nominal phrase ‘al médico’ and is, thus, redundant. Nevertheless, removing this pronoun would produce a wrong (that is, ungrammatical) sentence. So a coreference mechanism has to be allowed in order to tag this type of redundancies. This can be done also following the same mechanism used in Example 1 and 2, as shown in Example 3. The annotation associated to redundancy has been highlighted for convenience. The example only includes the section for terminals for the sake of space.

```
<corpus ...>
 <head>
  <!-- ... -->
  <meta>
   <name>spanish</name>
  </meta>
  <annotation>
   <!-- ... -->
   <feature domain="edge" name="label" type="coref">
    <value name="Anaph">Anaphoric</value>
   </feature>
  </annotation>
 </head>
 <body>
  <s xml:id="s1">
   <graph root="s1_ROOT" discontinuous="false">
    <terminals>
     <t xml:id="s1_t1"
       tiger2:corresp="spanish.example.maf.xml#wordForm2"/>
     <!-- Decir (Di-) = Tell -->
     <t xml:id="s1_t2"
       tiger2:corresp="spanish.example.maf.xml#wordForm3">
      <edge tiger2:type="coref" label="Anaph"
          tiger2:target="#s1_nt4"/>
      <!-- se (-se-) = to him -->
     </t>
     <t xml:id="s1_t3" tiger2:corresp="spanish.maf.xml#wordForm4"/>
     <!-- lo (-lo) = it -->
     <t xml:id="s1_t4" tiger2:corresp="spanish.maf.xml#wordForm6"/>
     <!-- a (a-) = to -->
     <t xml:id="s1_t5" tiger2:corresp="spanish..maf.xml#wordForm7"/>
     <!-- el (-l) = the -->
     <t xml:id="s1_t6" tiger2:corresp="spanish.maf.xml#wordForm8"/>
     <!-- médico = doctor -->
     <t xml:id="s1_t7" tiger2:corresp="spanish.maf.xml#wordForm9"/>
     <!-- . -->
    </terminals>
   <nonterminals>
    <!-- ... -->
   </nonterminals>
  </graph>
```

```
  </s>
 </body>
</corpus>
```

Example 3: SynAF-<TIGER2/>-conformant syntactic annotation of the Spanish sentence 'Díselo al medico.'

4.2 Semitic Languages

Semitic languages bring with them a particular set of challenges for modelling linguistic data. The following two subsections demonstrate how MAF and SynAF/<tiger/2> can work together to represent two problematic constructions: dependencies for object clitics and constituents for construct state compounds.

4.2.1 Arabic enclitic objects

Much like Romance languages, both Arabic and Hebrew have the ability to express pronominal objects as enclitic pronouns that are written together with the verb, forming one word form on the orthographic level [9], e.g. in Standard Arabic:

رأيت الملك
ra'aitu l-malik 'I saw the king'
see.PF.1.SG DEF-king

رأيته
ra'aitu-hu 'I saw him'
see.PF.1.SG-3.SG.M

Modelling a syntax tree of the dependencies between verb and object can be difficult and usually leads to make the design decision of tokenizing enclitic object pronouns as separate word forms. Using MAF word forms it is possible to keep such orthographic strings together while referring directly to constituent elements in the SynAF representation. The following representation gives a possible solution for a unified representation of the examples above, similar to the Spanish case.

```
<maf>
  <token xml:id="s1tk1">رأيت</token>        <!-- ra'aitu = I saw-->
  <token xml:id="s1tk2">ال</token>          <!-- l = the -->
  <token xml:id="s1tk3">ملك</token>         <!-- malik = king -->
  <token xml:id="s1tk4">.</token>           <!-- . -->
  <token xml:id="s2tk1">رأيت</token>        <!-- ra'aitu = I saw -->
  <token xml:id="s2tk2">ه</token>           <!-- hu = him -->
  <token xml:id="s2tk3">.</token>           <!-- . -->

  <wordForm xml:id="s1wf1" tokens="s1tk1">          <!-- ra'aitu -->
  <wordForm xml:id="s1wf2" tokens="s1tk2 s1tk3">    <!-- l-malik -->
  <wordForm xml:id="s1wf3" tokens="s1tk4">          <!-- . -->
  <wordForm xml:id="s2wf1" tokens="s2tk1 s2tk2">    <!--ra'aitu-hu -->
  <wordForm xml:id="s2wf2" tokens="s2tk3">          <!-- . -->
</maf>
```

Example 4: MAF representation of the Arabic clitic examples above.

```
<s xml:id="s1">
 <graph>
  <terminals>
   <!-- ra'aitu = I saw -->
   <t xml:id="s1_t1" tiger2:corresp="arabic.maf.xml#s1wf1">
    <edge xml:id="s1_e1" tiger2:target="#s1_t3" tiger2:type="dep"
     label="obj"/>
   </t>
   <!-- l = the -->
   <t xml:id="s1_t2" tiger2:corresp="arabic.maf.xml#s1tk2" />
   <!-- malik = king -->
   <t xml:id="s1_t3" tiger2:corresp="arabic.maf.xml#s1tk3">
    <edge xml:id="s1_e2" tiger2:target="#s1_t2" tiger2:type="dep"
     label="det"/>
   </t>
  </terminals>
  <nonterminals/>
 </graph>
</s>
<s xml:id="s2">
 <graph>
  <terminals>
   <t xml:id="s2_t1" tiger2:corresp="arabic.maf.xml#s2tk1" >
    <edge xml:id ="s1_e1" tiger2:target="#s2_t2" tiger2:type="dep"
     label="obj"/>
   </t>
   <t xml:id="s2_t2" tiger2:corresp="arabic.maf.xml#s2tk2" />
  </terminals>
  <nonterminals/>
 </graph>
</s>
```

Example 5: <tiger2/> representation of dependency trees for the Arabic examples above.

By pointing at both word forms and tokens, the SynAF representation can contain uniform syntax trees for both enclitic and separate objects.

4.2.2 Hebrew construct states

Both Hebrew and Arabic use a special left-headed construction similar to compounding [10] to form complex nouns. This construction has a number of special features. Most notably for the present discussion, it does not constitute an orthographic word form, but it allows only one article for the entire complex noun, placed between the head and the modifier (or before the last modifier in recursive complex nouns). The article, like all articles in both languages, is written together with the final modifier and does constitute one word form with it, but marks the entire construction as definite. The following examples from Hebrew illustrate the positioning of the relevant elements in definite and indefinite environments:

בית חולים
beit xolim 'hospital'
house sick.PL

בית החולים
beit ha-xolim 'the hospital'
house the-sick.PL

As a result of this configuration, a syntactic analysis may choose to regard the interposed article as attached to the entire complex noun around it (‘hospital’), rather than the modifier to which it is attached (‘sick’). A MAF/SynAF representation reflecting this analysis is given below. Note that the MAF representation contains a discontinuous word form to represent the word for ‘hospital’ and its lemma, as well as a word form uniting ‘sick’ with the orthographically attached article, but without a lemma (since ‘the sick’ has no lexical status as a lemma).

```
<maf>
  <token xml:id="s1tk1">בית</token>         <!-- beit = house -->
  <token xml:id="s1tk2">ה</token>           <!-- ha = the -->
  <token xml:id="s1tk3">חולים</token>       <!-- xolim = sick -->

  <!-- beit ...xolim -->
  <wordForm xml:id="s1wf1" lemma="בית חולים" tokens="s1tk1 s1tk3">
  <!-- ha-xolim -->
  <wordForm xml:id="s1wf2" tokens="s1tk2 s1tk3">
</maf>
```

Example 6: MAF representation of the Hebrew construct state examples above.

```
<s xml:id="s1">
 <graph>
  <terminals>
   <t xml:id="s1_t1" tiger2:corresp="hebrew.maf.xml#s1wf1"/>
   <!-- beit...xolim = hospital -->
   <t xml:id="s1_t2" tiger2:corresp="hebrew.maf.xml#s1tk2"/>
   <!-- ha = the -->
  </terminals>
  <nonterminals>
   <nt xml:id="s1_nt1" cat="NP">
    <edge tiger2:target="#s1_t2" tiger2:type="prim" label="DT"/>
    <!-- ha = the -->
    <edge tiger2:target="#s1_t1" tiger2:type="prim" label="HD"/>
    <!-- beit...xolim = hospital -->
   </nt>
  </nonterminals>
 </graph>
</s>
```

Example 7: <tiger2/> representation of a constituent tree for the definite construct state.

This way, the definite NP behaves in a usual manner by joining a noun with an article, and the complexity of the noun is hidden in the MAF representation. Another option is to model the entire graph within the SynAF level, as in the following representation.

```
<s xml:id="s1">
 <graph>
  <terminals>
   <t xml:id="s1_t1" tiger2:word="בית"/>     <!-- beit = house -->
   <t xml:id="s1_t2" tiger2:word="ה"/>       <!-- ha = the -->
   <t xml:id="s1_t3" tiger2:word="חולים"/>   <!-- xolim = sick -->
  </terminals>
  <nonterminals>
```

```
   <nt xml:id="s1_nt1" tiger2:type="construct" lemma="בית חולים">
    <edge tiger2:target="#s1_t1" tiger2:type="const" label="HD"/>
    <!-- beit = house -->
    <edge tiger2:target="#s1_t3" tiger2:type="const" label="MO"/>
    <!-- xolim = sick -->
   </nt>
   <nt xml:id="s1_nt2" tiger2:type="phrase" cat="NP">
    <edge tiger2:target="#s1_t2" tiger2:type="prim" label="DT"/>
    <!-- ha = the -->
    <edge tiger2:target="#s1_nt1" tiger2:type="prim" label="HD"/>
    <!-- beit...xolim = hospital -->
   </nt>
  </nonterminals>
 </graph>
</s>
```

Example 8: <tiger2/> representation of the definite construct state without a MAF layer.

In this representation there are two types of non-terminals and edges: a phrase node with primary dominance edges ('prim'), and a construct node with 'const' edges to its constituents. These must be declared in the annotation block and may carry different annotations, e.g. the lemma for construct states.

4.3 Zulu

4.3.1 Zulu agreement and writing system, encoding in MAF

The Bantu language family, of which Zulu is a member, is characterized by an agreement system based on noun classification [11]. The basic word order in Zulu is SVO, although this availability of grammatical agreement allows a fairly unrestricted word order. In Zulu, a verb hence usually contains a prefix that cross-references it to the subject noun. Zulu is a language written conjunctively, i.e. linguistic units are merged to morpho-phonetic units on the surface.

4.3.2 Cross-referencing noun phrases in verb phrases

In the example sentence shown in Table 1, the *wa-* prefix in the verb *wabuya* agrees with the subject noun *umfana*. Both are of noun class 1, which usually indicates that the noun is referring to a singular human. In Table 1, the first line shows the surface sentence, the second its underlying morphs, and the third line demonstrates its part-of-speech (or rather morph) categories.

Umfana		wabuya			kodwa	akayisizanga			
Um-	fana	wa-	-buy-	-a	kodwa	aka-	-yi-	-siz-	-anga
class 1 nom. marker	noun stem	subj-3rdcl1 prefix past tense	verb root	verb ending	CONJ	neg.subj-3rd-cl1 prefix	obj-3rd-cl9 prefix	verb root	verb negation marker past tense
	boy		return		but	(he)	her	assist	not

Table 1: A brief description of the Zulu sentence 'Umfana wabuya kodwa akayisizanga' (The boy returned but he did not assist her)

While subject-verb agreement is obligatory, the verb-object phrase only shows grammatical agreement in certain cases, for instance, when the referent of the object is known information in the discourse and therefore omitted, as illustrated in the example sentence. The object-affix *–yi-* in *akayisizanga* is an example of pronominal incorporation in the verb, since it refers to the class 9 noun *intombazane* (girl), a discourse topic that has already been established. Note that, although it is a dependent morpheme, this object affix carries the grammatical function of an object (see also [12] on *Kichaga*, another Bantu-language).

4.3.3 Inflection (negation and tense indication)

One way to negate a sentence in Zulu is to modify the verb. In this case, the sentence is negated by using two verbal affixes, i.e. the prefix *aka-* and the suffix *-anga* which also indicates past tense (*-i* would have marked present tense). In the first verb, *wa-buya*, past tense is indicated by the class prefix *wa-*; here, present tense would have been indicated by the prefix *u-*.

4.3.4 Compound sentences

The sentence in Table 1 is an example of a compound sentence that links clauses of equal importance. In this case, the co-ordination is achieved by means of the co-ordinating conjunction *kodwa* (but). Our suggested encoding in MAF (the token definitions are found in Example 9) reflects the morpho-syntactic facts: some of the units consisting of several morphs are formed on word level (see Example 10), while others form the predicate, i.e. a verbal phrase containing an agreement marker, an object, a verb stem and an inflectional element, cf. (Example 11).

```
<token join="right" xml:id="t1" form="Um"/>
<!-- um: noun class prefix class 1-->
<token xml:id="t2" form="fana"/>
<!-- fana: noun stem common (boy)-->
<token join="right" xml:id="t3" form="wa"/>
<!-- wa: subject prefix class 1-->
<token xml:id="t4" form="buy"/>
<!-- buy: verb root (return) -->
<token join="left" xml:id="t5" form="a"/>
<!-- -a: verb ending/extension -->
<token xml:id="t6" form="kodwa"/>
<!-- kodwa: conjunction (but)
<token join="right" xml:id="t7" form="aka"/>
<token join="right" xml:id="t8" form="yi"/>
<token xml:id="t9" form="siz"/>
<token join="left" xml:id="t10" form="anga"/>
<token xml:id="t11" form="."/>
```

Example 9: Excerpt of the MAF-encoding: token definitions

```
<wordForm xml:id="wordForm10" tokens="t1 t2" lemma="umfana">
 <wordForm tokens="t1">
  <fs>
   <f name="pos">
    <symbol value="morph.CP class.01 pers.01 num.sg"/>
   </f>
  </fs>
 </wordForm>
 <wordForm>
  <fs>
   <f name="pos"> <symbol value="morph.NSC"/> </f>
  </fs>
 </wordForm>
</wordForm>
```

Example 10: Excerpt of the MAF-encoding: morphs forming a noun

```
<!-- the wordform consists of four wordForms, the first two and -->
<!-- the last one are grammatical morphemes. -->
<!-- These will be put together in Synaf because the result is -->
<!-- a syntactic category rather than a word -->
<wordForm xml:id="wordForm40" tokens="#t7">
 <fs>
  <f name="pos">
   <symbol value="morph.CS class.01 pers.03 num.sg pol.neg"/>
  </f>
 </fs>
</wordForm>
<wordForm xml:id="wordForm41" tokens="#t8">
 <fs>
  <f name="pos">
   <symbol value="morph.CO class.09 pers.03 num.sg"/>
  </f>
 </fs>
</wordForm>
<wordForm xml:id="wordForm42" tokens="#t9" lemma="siza">
 <fs>
  <f name="pos"><symbol value="morph.VR_tr"/></f>
 </fs>
</wordForm>
<wordForm xml:id="wordForm43" tokens="#t10">
 <fs>
  <f name="pos"><symbol value="morph.neg" tense="past"/></f>
 </fs>
</wordForm>
```

Example 11: Excerpt of the MAF-encoding: units of the predicate

4.3.5 Encoding in SynAF

For space reasons, we focus on two of the phenomena described above: defining an anaphoric target in the second clause (see Example 12), and defining the predicate as a syntactic unit (a VP, see Example 13) formed by surface linguistic tokens, some of which are dependent morphs and others are free morphemes, i.e. lexical units.

```
<terminals>
 <t xml:id="s1_t1"
    tiger2:corresp="example.standoff.maf.xml#wordForm10"/>
 <!-- umfana -->
 <!-- ... -->

 <t xml:id="s1_t6" tiger2:type="PRO" function="SC">
  <!-- (umfana) -->
  <edge tiger2:type="coref" label="anaphoric" tiger2:target="#s1_t1"/>
 </t>
</terminals>
```

Example 12: Excerpts of our SynAF-encoding demonstrating the capability of the standard to represent anaphoric elements, non-existent on the surface.

```
<!--...-->
<!-- aka-yi-siz-anga -->
<t xml:id="s1_t7"
   tiger2:corresp="example.standoff.maf.xml#wordForm40"/>
<t xml:id="s1_t8"
   tiger2:corresp="example.standoff.maf.xml#wordForm41"/>
<t xml:id="s1_t9"
   tiger2:corresp="example.standoff.maf.xml#wordForm42"/>
<t xml:id="s1_t10"
   tiger2:corresp="example.standoff.maf.xml#wordForm43"/>
<!--...-->
</terminals>
<!--...-->
<non-terminals>
 <nt xml:id="s1_nt400" cat="VP01">   <!-- aka-yi-siz-anga -->
  <edge tiger2:type="prim" tiger2:target="#s1_t7"/>
   <edge tiger2:type="prim" tiger2:target="#s1_t8"/>
   <edge tiger2:type="prim" tiger2:target="#s1_t9"/>
   <edge tiger2:type="prim" tiger2:target="#s1_t10"/>
  </nt>
</non-terminals>
```

Example 13: Description of terminal units that together appear as a syntactic element (the predicate)

4.4 Chinese

The following encodings illustrate the treatment of an example sentence in Chinese. The sentence reads "她们正在学习古代汉语。" ("*They are currently studying classical Chinese.*"). It contains a total of 11 Chinese characters including the sentence-end full stop, which are segmented into 6 lexical units as terminals. Syntactically speaking, the string, like most other sentences in Chinese, exhibits an SVO construction, with "她们" (*They*) as the subject NP, "学习" (*studying*) as the verb and "古代汉语" (*classical*

Chinese) as the direct object NP. “正在” (*currently*) is analyzed as an adverb, forming part of the VP.

There is no explicit tense marking in the sentence. The progressive aspect is lexically expressed through the adverb “正在” (*currently*), reaffirming the fact that there is no morphological change for the Chinese verb. As a matter of fact, except for a few dependent morphemes indicating plurality, Chinese nouns do not have inflectional changes. In this sense, the attribute “lemma” in <terminals> below is not really necessary. In addition, the plural personal pronoun “她们” (*they*) is feminine, illustrating one of the very few personal pronouns carrying gender information. As a general rule, there is no gender marking for Chinese nouns. It is thus questionable whether gender marking should be a default attribute. In the current example, such marking is omitted for the sake of simplicity and clarity.

```
<s xml:id="s1">
 <graph root="s1_ROOT" discontinuous="true">
  <terminals>
   <t xml:id="s1_t1" tiger2:word="她们" lemma="她们" pos="PP"/>
   <t xml:id="s1_t2" tiger2:word="正在" lemma="正在" pos="RB"/>
   <t xml:id="s1_t3" tiger2:word="学习" lemma="学习" pos="VB"/>
   <t xml:id="s1_t4" tiger2:word="古代" lemma="古代" pos="JJ"/>
   <t xml:id="s1_t5" tiger2:word="汉语" lemma="汉语" pos="NN"/>
   <t xml:id="s1_t6" tiger2:word="。" lemma="。" pos="."/>
  </terminals>
  <nonterminals>
   <nt xml:id="s1_nt1" cat="NP">
    <edge tiger2:target="#s1_t1" tiger2:type="prim" label="HD"/>
     <!--她们-->
   </nt>
   <nt xml:id="s1_nt2" cat="VP">
    <edge tiger2:target="#s1_t2" tiger2:type="prim"/>
    <!-- 正在 -->
    <edge tiger2:target="#s1_t3" tiger2:type="prim" label="HD"/>
    <!-- 学习 -->
    <!-- labels can be ommited as in the next edge -->
    <edge tiger2:target="#s1_nt3" tiger2:type="prim" label="DO"/>
    <!-- NP -->
   </nt>
   <nt xml:id="s1_nt3" cat="NP">
    <edge tiger2:target="#s1_t4" tiger2:type="prim" label="AT"/>
    <!-- 古代 -->
    <edge tiger2:target="#s1_t5" tiger2:type="prim" label="HD"/>
    <!-- 汉语 -->
   </nt>
   <nt xml:id="s1_nt4" cat="S">
    <edge tiger2:target="#s1_t1" tiger2:type="prim" label="SB"/>
    <!--她们 -->
    <edge tiger2:target="#s1_nt2" tiger2:type="prim"/>
    <!-- VP -->
   </nt>
   <nt xml:id="s1_ROOT" cat="ROOT">
    <edge tiger2:target="#s1_nt2" tiger2:type="prim"/>
    <!-- VP -->
    <edge tiger2:target="#s1_t6" tiger2:type="prim"/>
    <!-- 。 -->
   </nt>
```

```
  </nonterminals>
 </graph>
</s>
```

Example 14: example sentence in Chinese

4.5 Korean

Korean is known to be an *agglutinative head-final* language, much like Japanese (see [13] and [14] for details).

4.5.1 Agglutination

Basic sentential segments, separated by whitespace, are called *eojeol* in Korean. Each eojeol is formed through a series of so-called *agglutination* processes. Each agglutination process concatenates a stem with a suffix, either nominal or predicate, and then this process may repeat, as in [[[타]verbStem-[쓰]pas]stem-[던]rx]eojeol.

미아가 탔던 차
mia.ga that.deon cha
Mia+NOM use+PAS+RX car
'car (which) Mia used to ride'

where NOM is a nominative case marker, PAS a past tense marker, and RX a relativizing (retrospective) suffix, called *adnominalizer*.

This string consists of three *eojeol*: (eo1) 미아가, (eo2) 탔던, and (eo3) 차. Some eojeol, such as (eo2) and (eo3), are word forms by themselves, whereas the first eojeol (eo1) is segmented into two word forms, 미아 'Mia' and –가, a nominative case marker. Each word form is then segmented into one or more tokens.

The process of segmenting a textual string into tokens or that of combining them into word forms can be represented in conformance to MAF:

```
<s xml:id="s1" lang="kr">미아가 탔던 차</s>
<maf>
  <token xml:id="s1_tk1">미아</token>
  <!-- mia: 'Mia' -->
  <token xml:id="s1_tk2" join="left">가</token>
  <!-- ga: nominative case marker -->
  <token xml:id="s1_tk3">타</token>
  <!-- tha: 'ride' -->
  <token xml:id="s1_tk4" join="overlap">ㅆ</token>
  <!-- t or ss: past tense makrer -->
  <!-- tk3 and tk4 form one syllable character, 탔: that or thass -->
  <token xml:id="s1_tk5" join="left">던</token>
  <!-- deon: relativizing suffix -->
  <token xml:id="s1_tk6">차</token>
  <!--  cha: 'car' -->
  <wordForm xml:id="s1_wf1" lemma="미아" tag="#pos.properName"
   tokens="s1_tk1" />
  <wordForm xml:id="s1_wf2" lemma="-가" tag="#pos.case_marker"
   tokens="s1_tk2" />
  <wordForm xml:id="s1_wf3" lemma="타다" form="탔던" tag="#pos.verb"
   tokens="s1_tk3  s1_tk4 s1_tk5" />
```

```
  <!-- The lemma 타다 consists of a stem 타 'ride' and a final verbal
   suffix 다. -->
  <wordForm xml:id="s1_wf4" lemma="차" tag="#pos.noun"
   tokens="s1_tk6"/>
</maf>
```

Example 15: Tokens and Word Forms in MAF

4.5.2 Head-final Feature

Korean is an SOV language with predicate forms at the end of the clause. The primary data given above shows a relative clause construction ending in a verb form 탔던 'used to ride' (relativized form) and followed by its nominal head 차 'car'. This syntactic formation can be represented as follows:

```
<s xml:id="s1">
 <graph>
  <terminals>
   <t xml:id="s1_t1" tiger2:corresp="korean.maf.xml#s1_wf1"/>
   <t xml:id="s1_t2" tiger2:corresp="korean.maf.xml#s1_wf2"/>
   <t xml:id="s1_t3" tiger2:corresp="korean.maf.xml#s1_wf3"/>
   <t xml:id="s1_t4" tiger2:corresp="korean.maf.xml#s1_wf4"/>
  </terminals>
  <nonterminals>
   <nt xml:id="s1_nt1" tiger2:type="eojeol" cat="postpositionPhrase">
    <edge xml:id="s1_e1" tiger2:type="noun" tiger2:target="#s1_t1"/>
    <edge xml:id="s1_e2" tiger2:type="case_marker"
     tiger2:target="#s1_t2"/>
   </nt>
   <nt xml:id="s1_nt2" tiger2:type="eojeol" cat="verb">
    <edge xml:id="s1_e3" tiger2:type="wordForm"
     tiger2:target="#s1_t3"/>
   </nt>
   <nt xml:id="s1_nt3" tiger2:type="clause" cat="relativeClause">
    <edge xml:id="s1_e11" tiger2:type="phrase"
     tiger2:target="#s1_nt1"/>
    <edge xml:id="s1_e12" tiger2:type="word" tiger2:target="#s1_nt2"/>
   </nt>
   <nt xml:id="s1_nt4" tiger2:type="eojeol" cat="word">
    <edge xml:id="s1_e4" tiger2:type="noun" tiger2:target="#s1_t4"/>
   </nt>
  </nonterminals>
 </graph>
</s>
```

Example 16: Relative Clause Construction in Korean

5 Generating <tiger2/> data automatically

One of the goals of the <tiger2/> development was not just to provide a new XML format, but also a human and machine readable metamodel and a processable API. For this purpose we used the Eclipse Modeling Framework (EMF). We generated the API in Java, because of its wide use and operating system interoperability. Since the core of the <tiger2/> metamodel is a general graph structure of nodes, edges and labels on both, it is possible, to access the <tiger2/> model with the usual graph processing mechanisms such as traversal etc.

Another part of the <tiger2/> development was its compatibility to the TigerXML model, i.e making sure that existing data in TigerXML can be converted to <tiger2/> without loss of information. To test this, we implemented a mapping from TigerXML to the <tiger2/> API. This allows us to import already existing data in TigerXML into the API and export it to <tiger2/>, and vice versa, though this may lead to information losses in some cases. Testing native TigerXML files and converting them back and forth, we can ensure no information is lost.

Another interesting prospect is converting data from other graph-based formats into <tiger2/>. We therefore used the SaltNPepper framework [15], a universal importer to convert a wide range of formats into each other. Pepper is a plug-in based converter which uses the intermediate metamodel Salt to make a direct conversion between several formats. Using SaltNPepper and the <tiger2/> API we created a module mapping <tiger2/> to the Salt metamodel which we plugged into the framework. This step has allowed us to benefit from all already existing SaltNPepper modules and to convert data e.g. from and into the CoNLL (http://ilk.uvt.nl/conll/#dataformat) format, the PAULA XML format [16], the GrAF format [17] and more.

The CoNLL format, is a field-based format largely used in international parsing evaluations. It is dependency-based, usually restricted to projective dependencies, with several fields such as FORM, LEMMA, CPOSTAG, POSTAG, FEATS, etc. A conversion to <tiger2/> is rather straightforward, with no need for **nt** elements (see Example 17 and 18).

1	il	il	CL	CLS	_	2	suj	_	_
2	mange	manger	V	V	_	0	root	_	_
3	une	un	D	DET	_	4	det	_	_
4	pomme	pomme	N	NC	_	2	obj	_	_
5	.	.	PONCT	PONCT	_	2	ponct	_	_

Example 17: CoNLL output for "he eats an apple"

```
<t form="il" lemma="il" cpostag="CL" postag="CLS" xml:id="1">
 <edge label="suj" target="2"/>
</t>
<t form="mange" lemma="manger" cpostag="V" postag="V"/>
<!-- ….-->
```

Example 18: Fragment of a possible <tiger2/> representation for CoNLL

When using the CoNLL format, some problems may arise from the fact that it does not follow the two level segmentation model of MAF (with tokens and word forms), leading to compound POS such as **P+D** for the agglutinate *des* ('de' + 'les' = *of the*).

In a second experiment, a dataset was converted from the Passage format [18] produced by the French parser FRMG [19][4] into the <tiger2/> format. Though there is no module for SaltNPepper supporting the Passage format at present, a dedicated <tiger2/> generator was constructed for this purpose. The Perl conversion script was easily developed, and, while it remains to test it on complex cases, it demonstrates the potential of <tiger2/> to encode relatively complex syntactic annotation formats.

The Passage format was developed for the French evaluation campaign also called Passage (http://atoll.inria.fr/passage/). Accordingly, the Passage format is currently produced by several French parsers, and is relatively rich in information. It is a successor of the EASy format, extended to better conform the recommendations of ISO TC37SC4, in particular with the MAF format (at the tokenization level with **T** and **W** elements) and with the use of ISO data categories. Passage is both constituency-based, with 6 kinds of chunks (called **G**) and dependency-based, with 14 kinds of relations anchored by either word forms **W** or chunks **G**. The **T** and **W** elements are straightforwardly converted into MAF elements and a correspondence (with attribute @corresp) is established at the level of <tiger2/> elements **t** towards the MAF word forms. The chunks become **nt** elements and the relations become **edge** elements either within **t** or **nt** elements. However, strictly speaking, Passage relations are not oriented (since they entail no explicit notion of governor and governee), but rolebased. Furthermore, the **COORD** relation (for the coordinations) is ternary and has 3 roles, namely **coordonnant** (coordinator), **coord-g** (left coordinated) and **coord-d** (right coordinated). To fit in the <tiger2/> metamodel, it was therefore needed to orient the relations (by choosing a governor and orienting the edge from the governee to its governor) and binarize the COORD relation (using the coordinator as governor). Fortunately, no information is lost in the process. This is achieved using a @**label** attribute on **edge**, which combines the name of the relation with a role name (such as the **SUJ-V_sujet** label for the subject in a SUJ-V relation). Constituency is represented by edges labelled **comp** within **nt** elements. Finally, the additional information carried by Passage **W** elements, such as **form**, **lemma** and **mstag** are moved to the MAF **wordForm** elements, with a conversion of the **mstag** flat feature structure (using tags) into a deep expanded feature structure (based on the FSR standard). Example 21 shows the original Passage fragment and Example 19 and 20 the corresponding representation in MAF and <tiger2/>.

[4] FRMG may be tried on line at http://alpage.inria.fr/parserdemo with the possible production of 3 formats (DepXML,Passage, CoNLL). The <tiger2/> format should be soon added, thanks to the script presented in this paper.

```
<maf>
 <token xml:id="E1T1">ce</token>             <!-- this -->
 <token xml:id="E1T2">matin</token>          <!-- morning -->
 <token xml:id="E1T3">,</token>              <!-- , -->
 <token xml:id="E1T4">il</token>             <!-- he →
 <token xml:id="E1T5">a</token>              <!-- has -->
 <token xml:id="E1T6">rapidemment</token>    <!-- quickly -->
 <token xml:id="E1T7">mangé</token>          <!-- eaten -->
 <token xml:id="E1T8">une</token>            <!-- an -->
 <token xml:id="E1T9">pomme</token>          <!-- apple -->
 <token xml:id="E1T10">rouge</token>         <!-- red -->
 <token xml:id="E1T11">et</token>            <!-- and -->
 <token xml:id="E1T12">une</token>           <!-- a -->
 <token xml:id="E1T13">poire</token>         <!-- pear -->
 <token xml:id="E1T14">.</token>             <!-- . -->
 <wordForm form="ce" lemma="ce" tokens="E1T1" xml:id="E1F1">
  <fs>
   <f name="pos"><symbol value="demonstrativeDeterminer"/></f>
   <f name="dem"><symbol value="plus"/></f>
   <f name="numberposs"><symbol value="minus"/></f>
   <f name="gender"><symbol value="masc"/></f>
   <f name="number"><symbol value="sg"/></f>
  </fs>
 </wordForm>
 <!-- …. -->
</maf>
```

Example 19: Fragment of the MAF part (sample_passage.maf.xml) produced from PASSAGE

```
<?xml version="1.0" encoding="UTF-8"?>
<corpus xmlns="http://korpling.german.hu-berlin.de/tiger2/V2.0.5/"
 xmlns:tiger2="http://korpling.german.hu-berlin.de/tiger2/V2.0.5/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://korpling.german.hu-berlin.de/tiger2/V2.0.5/
 http://korpling.german.hu-berlin.de/tiger2/V2.0.5/Tiger2.xsd">
 <head>
  <meta>
   <name>conversion from Passage format</name>
   <author>passage2tiger2.pl</author>
   <date>2012-08-28 18:25:11</date>
   <format>PASSAGE format with French tagset</format>
  </meta>
  <annotation>
   <external corresp="PassageTAGSET.xml"/>
  </annotation>
 </head>
 <body>
  <s xml:id="s1">
   <graph>
    <terminals>
     <!-- this-->
     <t tiger2:corresp="sample_passage.maf.xml#E1F1"
      xml:id="E1F1"/>
     <!-- morning -->
     <t tiger2:corresp="sample_passage.maf.xml#E1F2" xml:id="E1F2">
```

```
      <edge label="MOD-V_modifieur" tiger2:target="E1F7"
       tiger2:type="prim"/>
     </t>
     <!-- ... -->
     <!--eaten -->
     <t tiger2:corresp="sample_passage.maf.xml#E1F7"
      xml:id="E1F7"/>
     <!-- an-->
     <t tiger2:corresp="sample_passage.maf.xml#E1F8"
      xml:id="E1F8"/>
     <!-- apple-->
     <t tiger2:corresp="sample_passage.maf.xml#E1F9" xml:id="E1F9">
      <edge label="COORD_coord-g" tiger2:target="E1F11"
       tiger2:type="prim"/>
     </t>
     <!-- red -->
     <t tiger2:corresp="sample_passage.maf.xml#E1F10"
      xml:id="E1F10">
      <edge tiger2:target="E1F9" label="MOD-N_modifieur"
       tiger2:type="prim"/>
     </t>
     <!-- and -->
     <t corresp="sample_passage.maf.xml#E1F11" xml:id="E1F11">
      <edge label="COD-V_cod" tiger2:target="E1F7" tiger2:type="prim"/>
     </t>
     <!-- a -->
     <t corresp="sample_passage.maf.xml#E1F12" xml:id="E1F12"/>
     <!-- pear -->
     <t corresp="sample_passage.maf.xml#E1F13" xml:id="E1F13">
      <edge label="COORD_coord-d" tiger2:target="E1F11"
       tiger2:type="prim"/>
     </t>
     <!-- ... -->
    </terminals>
    <nonterminals>
     <nt cat="GN" xml:id="E1G1">
      <edge label="comp" tiger2:target="E1F1" tiger2:type="prim"/>
      <edge label="comp" tiger2:target="E1F2" tiger2:type="prim"/>
     </nt>
     <!-- ... -->
    </nonterminals>
   </graph>
  </s>
 </body>
</corpus>
```

Example 20: Fragment of the <tiger2/> part (sample_passage.tiger2.xml) produced from PASSAGE and relying on the MAF part and an external PASSAGE tagset (PassageTAGSET.xml) , which specifies the possible features (pos, cat, lemma, form, …) and values (in particular for the edge labels and for the partof-speech

```
<Document dtdVersion="2.0" xmlns:xlink="http://www.w3.org/1999/xlink">
 <Sentence id="E1" trust="100">
  <T id="E1T1">ce</T>
  <T id="E1T2">matin</T>
```

```
  <G id="E1G1" type="GN">
   <W id="E1F1" pos="demonstrativeDeterminer" lemma="ce" form="ce"
    tokens="E1T1" mstag="wh.minus dem.plus numberposs.minus gender.masc
    number.sg"/>
   <W id="E1F2" pos="commonNoun" lemma="matin" form="matin"
    tokens="E1T2" mstag="person.3 wh.minus time.arto hum.minus
    gender.masc number.sg"/>
  </G>
  <!-- ... -->
  <R type="MOD-V" id="E1R5">
   <modifieur ref="E1F2"/>
   <verbe ref="E1F7"/>
  </R>
  <R type="COORD" id="E1R8">
   <coordonnant ref="E1F11"/>
   <coord-g ref="E1F9"/>
   <coord-d ref="E1F13"/>
  </R>
 </Sentence>
</Document>
```

Example 21: Fragment of the original PASSAGE file used to produce the MAF and <tiger2/> parts

6 Next steps

Even if the <tiger2/> proposal, building upon the already quite mature TigerXML format, does already present all the technical features to cover the characteristics of the SynAF metamodel, making it an ISO standard, usable for the largest possible scientific community will require some additional efforts. In particular, beyond the provision of simple examples, we will need to put <tiger2/> to the test by confronting it with large-scale treebanks as they are available in the various languages tackled in this paper. In particular, this will bring more ideas as to the data categories that should be added to ISOCat in order to get a fully-fledged environment for the description of annotation schemes for syntactic data.

References

[1] ISO 24615:2010. Language resource management – Syntactic annotation framework (SynAF).

[2] König, E. & Lezius, W. (2000). The TIGER language – A Description Language for Syntax Graphs. In: *Proceedings of COLING 2000*, 1056–1060.

[3] Leech G. N., Barnett, R. & Kahrel, P. (1995). *EAGLES Final Report and Guidelines for the Syntactic Annotation of Corpora*. EAGLES Document EAGTCWG-SASG. Pisa, Italy.

[4] Ide, N. & Romary, L. (2003). Encoding Syntactic Annotation. In: A. Abeillé (Ed.) *Treebanks*, Springer [available online at http://hal.archives-ouvertes.fr/hal-00079163].

[5] ISO/FDIS 24611. Language resource management – Morpho-syntactic annotation framework (MAF).

[6] Alarcos-Llorach, E. (2004[1999]). *Gramática de la lengua española*. Madrid: Espasa Libros, S.L.U.

[7] Real Academia Española (2009). *Nueva gramática de la lengua española*. Madrid: Espasa Libros, S.L.U.

[8] Pineda, L. & Meza, I. (2005). The Spanish pronominal clitic system. In: *SEPLN (Revista de la Sociedad Española para el Procesamiento del Lenguaje Natural)* 34 [available online at http://www.sepln.org/revistaSEPLN/revista/34/06.pdf (visited on 28 August 2012)].

[9] Ryding, K. C. (2005). *A Reference Grammar of Modern Standard Arabic*. Cambridge: Cambridge University Press.

[10] Borer, H. (2008). Compounds: The View from Hebrew. In: R. Lieber & P. Štekauer (eds.), *The Oxford Handbook of Compounds*. Oxford: Oxford University Press, 491–511.

[11] Nurse, D. & Philippson, G. (2003). *The Bantu Languages*. London: Routledge.

[12] Bresnan, J. (2000). *Lexical-Functional Syntax*. Blackwell Publishing: MA.

[13] Chang, Suk-Jin (1996). *Korean*. Amsterdam: John Benjamins Publishing Co.

[14] Sohn, H.-M. (1999). *The Korean Language*. Cambridge: Cambridge University Press.

[15] Zipser, F., & Romary, L. (2010). A Model Oriented Approach to the Mapping of Annotation Formats Using Standards. In *Proceedings of the Workshop on Language Resource and Language Technology Standards, LREC 2010*. Malta, 7-18.

[16] Dipper, S. (2005). XML-based Stand-off Representation and Exploitation of Multi- Level Linguistic Annotation. In *Proceedings of Berliner XML Tage 2005 (BXML 2005)*. Berlin, Germany, 39-50.

[17] Ide, N., & Suderman, K. (2007). GrAF: A Graph-based Format for Linguistic Annotations. In *Proceedings of the Linguistic Annotation Workshop 2007*, Prague, 1-8.

[18] Paroubek, P., Villemonte de la Clergerie, E., Loiseau, S., Vilnat, A., Francopoulo. G (2009). The PASSAGE syntactic representation. In : *7th International Workshop on Treebanks and Linguistic Theories (TLT7)*. Groningen.

[19] Villemonte de la Clergerie, É. (2010). Convertir des dérivations TAG en dépendances. In *ATALA, Ed., 17e Conférence sur le Traitement Automatique des Langues Naturelles - TALN 2010*.

Effectively long-distance dependencies in French : annotation and parsing evaluation

Marie Candito[⋆] and Djamé Seddah[⋆◇]

[⋆] Alpage (Univ. Paris Diderot & INRIA), 175 rue du Chevaleret, 75013 Paris, France
[◇] Univ. Paris Sorbonne, 28, rue Serpente, 75006 Paris, France
marie.candito@linguist.jussieu.fr, djame.seddah@paris-sorbonne.fr

Abstract

We describe the annotation of cases of extraction in French, whose previous annotations in the available French treebanks were insufficient to recover the correct predicate-argument dependency between the extracted element and its head. These cases are special cases of LDDs, that we call *effectively* long-distance dependencies (eLDDs), in which the extracted element is indeed separated from its head by one or more intervening heads (instead of zero, one or more for the general case). We found that extraction of a dependent of a finite verb is very rarely an eLDD (one case out of 420 000 tokens), but eLDDs corresponding to extraction out of infinitival phrase is more frequent (one third of all occurrences of accusative relative pronoun *que*), and eLDDs with extraction out of NPs are quite common (2/3 of the occurrences of relative pronoun *dont*). We also use the annotated data in statistical dependency parsing experiments, and compare several parsing architectures able to recover non-local governors for extracted elements.

1 Introduction

While statistical parsers obtain high overall performance, they exhibit very different performance across linguistic phenomena. In particular, most statistical parsers perform poorly on long-distance dependencies (LDDs), which, though rare, are important to fully recover predicate-argument structures, which are in turn needed for semantic applications of parsing. Poor performance on LDDs is known of English statistical parsers, even though the training data does contain information for resolving unbounded dependencies (the Penn Treebank, or the specific dataset evaluated by Rimell et al. [17]). For French, the situation is worse, since the usual training data, the French Treebank (Abeillé and Barrier [1]), is a surface syntagmatic treebank that does not contain indications of LDDs : extracted elements bear a grammatical function, but no annotation indicates their embedded head. Hence syntagmatic stochastic French parsers cannot capture LDDs. Concerning dependency parsing, French dependency parsers can be learnt on the DEPFTB, resulting

from the automatic conversion of the FTB into projective dependency trees (Candito et al. [3]). But we will show that this automatic conversion leads to wrong dependencies for particular cases of LDDs - cases we call effectively-long-distance dependencies (eLDDs), in which the fronted element is extracted from an actually embedded phrase -, and thus statistical parsers learnt on the DEPFTB are unable to recover such cases correctly.

In this paper, we describe the manual annotation, performed on the FTB and the Sequoia treebank (Candito and Seddah [5]), of the correct dependencies in eLDDs, leading to non-projective dependency treebanks. We then evaluate several dependency parsing architectures able to recover eLDDs.

2 Target linguistic phenomena

Extraction is a syntactic phenomena, broadly attested across languages, in which a word or phrase (the extracted element) appears in a non-canonical position with respect to its head. For a given language, there are well-defined contexts that involve and licence an extraction. In English or French, the non-canonical position corresponds to a fronting of the extracted element.

A first type of extraction concerns the fronting of the dependent of a verb, as in the four major types topicalization (1), relativization (2), questioning (3), it-clefts (4), in which the fronted element (in italics) depends on a verb (in bold):

(1) *À nos arguments*, (nous savons que) Paul **opposera** les siens.
To our arguments, (we know that) Paull will-oppose his.

(2) Je connais l' homme *que* (Jules pense que) Lou **épousera**.
I know the man that (Jules thinks that) Lou will-marry.

(3) Sais-tu *à qui* (Jules pense que) Lou **donnera** un cadeau?
Do-you-know *to whom* (Jules thinks that) Lou will-give a present?

(4) C' est Luca *que* (Jules pense que) Lou **épousera**.
It is Luca that (Jules thinks that) Lou will-marry.

Since transformational grammar times, any major linguistic theory has its own account of these phenomena, with a vocabulary generally bound to the theory. In the following we will use 'extracted element' for the word or phrase that appears fronted, in non canonical position. One particularity of extraction is 'unboundedness': stated in phrase-structure terms, within the clause containing the extraction, there is no limit to the depth of the phrase the extracted element comes from. In dependency syntax terms, for a fronted element f appears either directly to the left of the domain of its governor g, or to the left of the domain of an ancestor of g. We focus in this paper on the latter case, that we call **effectively** long-distance dependencies (**eLDD**). In examples (1) to (4), only the versions with the material in brackets are eLDDs.

In French, another case of eLDD is when a PP is extracted from a predicative complement either nominal or adjectival:

(5) la mélancolie à laquelle il est enclin
the melancholy to which he is prone (*'the melancholy he is prone to'*)

Other cases of eLDDs arise for some PPs with preposition *de*, which under some well-studied conditions (see for instance Godard [8]) can be extracted from subject or direct object NPs, as in (6).

(6) un échec *dont* (Léo me dit que) les *causes* sont bien connues
a failure of-whom (Léo to-me says that) the causes are well known
'a failure whose causes (Leo tells me) are well known'

Those *de*-phrases are precisely the ones that can be cliticized, with the anaphoric clitic *en* (*of-it*) appearing on the verb governing the NP, as in (7).[1]

(7) Tu *en* connais bien les **raisons**
you of-it know well the reasons 'You know well the reasons for it'

To sum up, eLDDs comprise any case of extraction out of predicative complements, nominal subjects or objects (examples (5) to (7)), and cases of extraction of a dependent of a verb (examples (1) to (4)) only when involving intervening heads, (such as *to think* in these examples).

3 Target French Treebanks

3.1 French treebank and Sequoia treebank

Our objective is to obtain a dependency treebank for French with correct governors for extracted elements. This treebank will thus contain non-projective links. We perform our annotation of cases of extraction on two treebanks :

- the French Treebank (Abeillé and Barrier [1]) (hereafter FTB), a constituency treebank made of 12351 sentences[2] from the national newspaper *Le Monde*
- the Sequoia treebank (Candito and Seddah [5]), an out-of-domain corpus annotated following the FTB's annotation scheme. It contains roughly 1000 sentences from the French wikipedia, 1000 sentences from the medical domain (from medicines' marketing authorization reports from the European Medecine Agency), 500 sentences from Europarl and 500 sentences from the regional newspaper *L'Est Républicain*.

These are constituency treebanks, in which the dependents of verbs are labeled with a grammatical function, since a given structural position may correspond to different grammatical relations. Candito et al. [3] describe a tool for the automatic

[1] Note that in that case, the dependency between the clitic and the noun is bounded, but we still consider it a eLDD, because the clitic appears locally to the head (the verb) of the NP it depends on.

[2] As distributed in 2007. The current release has around 4000 additional sentences.

conversion of such constituency trees into surface dependency trees.[3] We briefly describe below this procedure, and detail the incorrect result obtained for eLDDs.

3.2 Automatic conversion to surface dependencies

The conversion procedure is based on the classic technique of head propagation rules, proposed for English by Magerman [10], and outputs projective surface dependency trees (each token has exactly one governor, except the root) : (i) Nodes in phrase-structure trees are annotated with their lexical head, using head-propagation rules, that state how to find the syntactic head in the right-hand side of a CFG rule; (ii) Using the lexical heads, bilexical dependencies are extracted. If the constituent node for the dependent bears a functional label, it is used as the label of the dependency; (iii) Remaining unlabeled dependencies are labeled using heuristics.

With that technique, output dependency trees are necessarily projective, and non-local dependents are wrongly attached to the local lexical head, as exemplified in Figure 1 in which the accusative relative pronoun *que* is wrongly attached to the local head *semblaient* (*seemed*) instead of its actual syntactic head *partager* (*to share*). A crucial point here is that a wrong dependency arises only for LDDs that are eLDDs. For instance from a simplified version of tree (a), without the raising verb, we would obtain the correct dependency between *que* and the verb *partager*.[4]

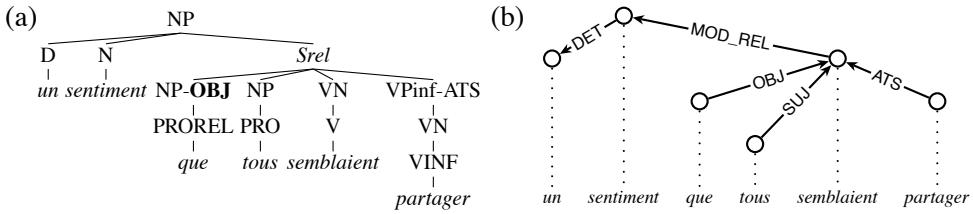


Figure 1: Left: An NP as annotated in the original French Treebank scheme (for *a feeling that (they) all seemed (to) share*). Right: corresponding automatically derived dependency tree, with wrong governor for the wh-word *que*.

4 Manual annotation

4.1 Selection of occurrences to annotate

One major difficulty to annotate extractions is that though ubiquitous in the linguistic literature, they are quite rare in actual texts. Further, some cases like topicalization (cf. example (1) above), involve structural clues only, and no lexical clue, namely no *wh*-words. Topicalization does exist in French, but is much rarer than

[3] Included in the BONSAI toolkit (http://alpage.inria.fr/statgram/frdep/fr_stat_dep_parsing.html).

[4] Note that the anaphoric relation between the pronoun and its antecedent is then trivial to recover, provided the relative clause is correctly attached.

relativization, it-clefts or questioning, and is restricted to the extraction of prepositional phrases. Hence, because we could not afford to scan the whole treebank to look for extractions, and also because it is unclear whether such dependencies can be recovered with current parsing techniques, we chose as a first step to focus on words known to be likely to involve an extraction, namely the clitic *en* and wh-words (relative and interrogative pronouns and determiners). This allows to capture the cases exemplified in section 2, except topicalization.

We examined each occurrence of wh-word and of the clitic *en*, and annotated the correct dependency in case of non-local dependency.

4.2 Annotation scheme and methodology

4.2.1 Functional paths

We chose to annotate non-locality using *functional paths*, made of sequences of dependency labels, as proposed in the LFG framework. We were inspired by Schluter and van Genabith [18], who have used them to annotate a manually modified version of half the French Treebank.[5] Before formalizing this notion, let us take as example the automatically-derived dependency tree in Figure 1, in which the relative pronoun *que* is wrongly attached to the local governor *semblaient*. The non-local governor is *partager*. We define the functional path for *que* to be the sequence of labels appearing on the path between *que* and its correct governor, namely *OBJ.ATS*, which can be read as *"'que' should be the OBJ of the ATS of its local governor"*.

More generally, let d be a lexical item that should depend on a non-local governor nlg. Let ADT be the dependency tree obtained by automatic conversion from the source constituency tree, and CDT the correct dependency tree that we target. Since nlg is non-local, the tree ADT contains a wrong dependency $lg \xrightarrow{l_0} d$, while CDT contains $nlg \xrightarrow{l_0} d$ (we suppose here that the label l_0 is correct is ADT). For such cases, we define a functional path as the sequence of labels $l_0.l_1....l_n$ that appear, in the incorrect tree ADT, on the path between the dependent d and its non-local governor nlg.

The manual annotation consists in making functional paths explicit : in the previous formalization, it amounts to modifying ADT into ADT', by labeling the dependency $lg \xrightarrow{l_0} d$ with the functional path as label : $lg \xrightarrow{l_0.l_1...l_n} d$. Note that eLDDs are exactly the cases involving a functional path of length > 1 instead of a simple functional tag (which can be regarded as a functional path of length 1).

4.2.2 Automatic interpretation of functional paths

This kind of annotation can be used in a straightforward way to recover the correct governor of extracted elements. We give in figure 2 the algorithm used to interpret functional paths as the indication of non-local dependencies : it changes a tree

[5] But these authors obtain only 65 cases in total, suggesting the linguistic constructions covered are few. Note we chose to use reverse functional paths, for easier reading.

containing functional paths into a corresponding tree, in which dependents that are non-local, are attached to their non-local governors.

0 • $ADT' \leftarrow ADT$
1 • while ADT' contains a dependency a of the form $lg \xrightarrow{l_0.l_1...l_n} d$, with $n > 0$, do
2 • $i \leftarrow n, g_i \leftarrow lg$
3 • while $i > 0$ do
4 • $G \leftarrow$ nodes x such as $g_i \xrightarrow{l_i} x$ and $x \neq d$
5 • if $G \neq \emptyset$, choose the left-most most appropriate node x, and set $g_{i-1} \leftarrow x$
6 • else, continues to next element in while 1
7 • $i \leftarrow i - 1$
8 • replace in ADT' the dependency a by $g_0 \xrightarrow{l_0} d$

Figure 2: Algorithm to interpret functional paths : find the non-local governors and change the dependency tree accordingly

If we go back to the example of Figure 1, the manual annotation applied to tree (b) is given in tree (c) in Figure 3. The interpretation of the sole functional path in tree (c) outputs the tree (d), which is non-projective.

The functional paths can be inconsistent with respect to the tree they appear in. This results in obtaining an empty set at line 4 in Figure 2. During the manual annotation phase, such cases can be used as warnings for a wrong functional path. When using the procedure in the parsing phase (see section 5), such functional paths are simply discarded and the label $l_0.l_1...l_n$ is replaced by l_0.

Further, the functional paths can be ambiguous. This is the case when the set G obtained at line 4 contains more than one element, at any point in the functional path. In these cases, the procedure prefers verbal governors over any other POS, and leftmost governor in case of remaining ambiguity.

This procedure is similar to the deprojectivization procedure defined by Nivre and Nilsson [15], when these authors use the encoding scheme they called 'Head'. More precisely, both procedures are equivalent in case of a functional path of length 2. We needed to define a procedure able to cope with paths of arbitrary length, in order to interpret any manually annotated functional path.

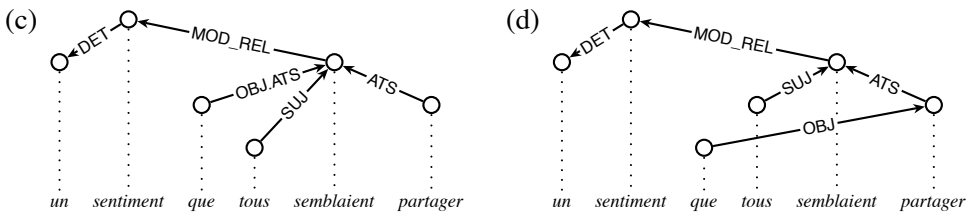


Figure 3: Left: Tree (c) is a modification of tree (b), with (manually) annotated functional path for the extracted element *que*. Right: Tree (d) is the output (non-projective) tree after using the algorithm Figure 2 to interpret functional paths.

4.2.3 Annotation methodology

Because the original treebanks we use have been annotated in constituency format, we chose to annotate the above-mentioned functional paths in the constituency trees, in order to retain the property that the dependency treebank can be automatically converted from the constituency trees. So, for the example of Figure 1, we annotate the functional path *OBJ.ATS* on the *NP* node that dominates the relative pronoun : NP-OBJ is replaced by NP-OBJ.ATS, then the usual constituency-to-dependency conversion produces the left tree of Figure 3, and the functional path interpretation procedure produces the right tree of Figure 3.[6]

We performed manual annotation of functional paths on a bracketed version of the French Treebank (called the FTB-UC by (Candito et al. [3])), and on the bracketed version of the Sequoia Treebank, using the WordFreak tool (Morton and LaCivita [13]), customized to handle the relevant sets of POS, non terminals and functional labels. The annotation for the words *en* and *dont* were performed independently by two annotators using WordFreak, and adjudicated by an expert annotator. The annotation for all the other wh-words were directly performed by a single expert annotator, because they potentially involve longer dependencies than for the word *en*, hence requiring to define more difficult (longer) functional paths. Then we applied the functional path interpretation algorithm (Figure 2), to obtain dependency versions of the FTB and the SEQTB with correct governors in case of eLDD. The resulting annotations are freely available.[7]

4.3 Quantitative characteristics

The resulting annotation provides a picture of the prevalence of extraction phenomena for a corpus of journalistic text (FTB) and for a corpus with mixed genres (the Sequoia corpus). We give various numbers of tokens for the annotated data in table 1, for the FTB, the SEQTB, and for the concatenation of both corpora.

The first observation is that the cases of extraction leading to eLDDs are very rare : for the whole FTB + SEQTB corpus, 0.16% of the tokens received a non-local governor (i.e. a functional path of length > 1). Further, more than 80% of eLDDs have a functional path of length 2, namely are "not-so-long" distance dependencies.

Focusing on projectivity, we see that only around two thirds of the eLDDs are non-projective (359 out of 618 dependencies). This is because most eLDDs are extractions from a subject NP, as in (6) (noted with functional path 'DEP.SUJ' : the extracted element is the dependent of the subject of the local (wrong) head). In

[6] We are aware that such an annotation task supposes a very good understanding of both the linguistic phenomena at play, and of the ad-hoc conversion to dependency procedure. Yet this has the advantage, over more traditional annotation using coindexed traces, that annotation on bracketed constituency trees is easier than on dependency structure.

[7] The SEQTB with correctly annotated eLDDs is available at https://www.rocq.inria.fr/alpage-wiki/tiki-index.php?page=CorpusSequoia. The FTB with non-local dependencies is available on request, provided you have the license for the FTB.

general, no dependent of the local verbal head intervenes between the subject and the extracted element (*en* or *dont*) as in (6), projectivity is preserved.[8]

	Number of tokens					
Treebank	total	with eLDD (%)	fpl=2	fpl=3	fpl>3	non projective
FTB	350931	555 (0.16 %)	466	69	20	317
SEQTB	69238	63 (0.09 %)	47	13	3	42
FTB + SEQTB	420169	618 (0.15%)	513	82	23	359

Table 1: For the FTB and the SEQTB, total number of tokens, tokens with non-local dependency, i.e. with length of functional path (fpl) > 1, tokens with fpl=2, fpl=3, fpl > 3; Number of tokens with fpl>1 that have a non projective dependency.

If we focus on the top three lexical items that exhibit eLDDs, we obtain the accusative relative pronoun *que*, the genitive relative pronoun *dont* and the anaphoric clitic *en*. As can be seen in table 2, these three lemmas totalize 570 out of the 618 cases of annotated eLDD in the FTB + SEQTB treebanks.[9] Note that though we saw eLDDs are very rare, these particular three lemmas have a substantial proportion of eLDD occurrences, especially *dont* (65.7% of its occurrences).

The most frequent element extracted from a verbal phrase is the relative pronoun *que*. A striking observation is that for all the 152 eLDD cases involving *que*, the embedded phrase *que* originates from is infinitival (as in the example Figure 3), and none is a finite clause.[10]

However, though the relative pronoun *dont* and the clitic *en* can either depend on verbs, nouns or adjectives, the majority of eLDDs are cases of extraction out of NPs. More precisely, out of subject NPs for *dont* (251 cases of functional paths DEP.SUJ out of 329) and out of object NPs for *en* (51 cases of functional paths DEP.OBJ, out of 89). The other prevalent case for clitic *en* is the extraction out of predicative complement (24 cases of functional paths DEP.ATS).

5 Parsing experiments

In this section we list and evaluate various parsing strategies able to output eLDDs. Though the overall parsing performance is unlikely to be affected by this parameter (given the very small amount of annotated eLDDs), it seems interesting to focus on attachment scores for the specific tokens that do often trigger eLDDs.

We relate experiments both using the "**local**" dependency trees, namely trees automatically converted from constituency trees without functional paths, and us-

[8]Non projectivity also arises when extracting a more deeply embedded dependent within the subject NP, as in *"une entreprise* dont *la moyenne d'âge des* **salariés** *dépasse 40 ans"* (*a company of-which the average of age of the employees is over 40*).

[9]Other cases concern (i) extractions of prepositional phrases (pied-piping), such as in example 3, for which the token that will bear the eLDD is the head preposition of the PP (*à* in example 3) or (ii) rare cases of extraction of NPs with wh-determiner, for which the noun bears the eLDD.

[10]We found no extraction out of an embedded finite clause in the FTB, and one in the SEQTB, in which the extracted element is a PP.

		Number of occurrences in FTB + SEQTB				
Lemma	POS	total	local gov	non-local gov	Top-3 most frequent fct paths	
que	relative pronoun	616	464	152 (32,8%)	OBJ.OBJ	77
					OBJ.OBJ.OBJ	22
					OBJ.OBJ.DE_OBJ	19
dont	relative pronoun	501	172	329 (65.7%)	DEP.SUJ	251
					DEP.OBJ	29
					DEP.ATS	27
en	accusative clitic	411	322	89 (21,7%)	DEP.OBJ	51
					DEP.ATS	24
					DEP.SUJ	11

Table 2: Statistics for the three lexical items having a non-local governor most frequently: total number of occurrences, number with and without local governor, and top-three most frequently-annotated functional paths.

ing the "**non local**" dependency trees, which have corrected dependencies for eLDDs (obtained via the procedure described in section 3.1). The local trees are projective, whereas the non local trees contain a few non projective links (two thirds of the eLDDs are non projective, cf. section 4.3).

We use the usual split for the 2007 version of the FTB (1235, 1235 and 9881 sentences for test, development and training sets respectively). For evaluation, we use as test sets the whole SEQTB on top of the usual FTB development and test sets.[11] In all our experiments the predicted parses are evaluated against the *non local* (pseudo-)gold dependency trees (while for training, we sometimes use the *local* dependency trees). We provide unlabeled and labeled attachment scores for all tokens except punctuation, and for the three lexical items that have a non local governor most frequently (the three lemma+POS pairs of Table 2).

We use MaltParser (Nivre et al. [14]), version 1.7 and MSTParser (McDonald [11]), version 0.5.0.[12] For the features, we use the best ones from a benchmarking of dependency parsers for French (Candito et al. [4]), except we removed unsupervised word clusters as features.[13] For MaltParser, we always use the arc-eager parsing algorithm, that can provide projective trees only.[14] For MSTParser, we either use order 1 factors and the Chu-Liu-Edmonds algorithm, that can output non projective trees, or order 2 factors, with the Eisner algorithm, restricted to projective trees. We also test pseudo-projective parsing (Nivre and Nilsson [15]), where non projective trees are transformed into projective ones, with label marking to allow the reverse the graph transformation: we use MaltParser to obtain projectivized versions of the non local trees, then either MaltParser or MSTParser to train a (projective) parser on these, and MaltParser again to de-projectivize the obtained

[11] As we did no parameter tuning, we did not reserve a test set for final test.

[12] Available at http://www.maltparser.org and http://sourceforge.net/projects/mstparser.

[13] POS are predicted using MORFETTE (Chrupała et al. [6]), and lemmas and morphological features are predicted using the BONSAI toolkit.

[14] Experiments with the swap algorithms showed degraded performance, supposedly due to the feeble amount of non-projective links.

predicted parses.[15]

			SEQTB			FTB dev			FTB test		
Parser	Type of training trees	Parsing algo	LAS	UAS	UAS on the 216 *que*, *en* *dont*	LAS	UAS	UAS on the 126 *que*, *en* *dont*	LAS	UAS	UAS on the 174 *que*, *en* *dont*
MALT	local	arc-eager	85.0	88.0	57.9 (125)	86.6	89.0	55.2 (75)	87.3	89.6	54.0 (94)
MALT	pproj	arc-eager	84.9	88.0	77.3 (167)	86.7	89.1	77.2 (105)	87.4	89.7	76.4 (133)
MST	non local	o1 np	85.0	88.2	71.8 (155)	86.5	89.2	80.2 (109)	87.3	89.9	80.5 (140)
MST	local	o2 proj	85.6	88.9	56.0 (121)	87.6	90.3	58.8 (80)	88.2	90.8	56.3 (98)
MST	non local	o2 proj	85.6	89.0	68.1 (147)	87.5	90.2	74.3 (101)	88.1	90.7	73.0 (127)
MST	pproj	o2 proj	85.7	89.1	71.3 (154)	87.6	90.3	80.9 (110)	88.4	91.0	78.7 (137)

Table 3: Parsing performance for malt or mst, with training on either local, non local, or projectivized (*pproj*) trees.

If we look at the overall performance, we observe, as usual, that MSTParser order 2 performs better than MaltParser. Further, training on the local, non local or projectivized trees has practically no impact (differences are not significant). When focusing on the lexical items that are likely to exhibit a non local dependency, the picture is quite different. First, it can be noted that using the non projective decoding for MSTParser, because it implies using order 1 factors, is not worth: the gain obtained for the eLDDs is too small to counterbalance the drop in performance when using order 1 instead of order 2 factors. Second, when comparing performance across the various training trees types, we note that pseudo-projectivization performs best on eLDDs. Moreover performance on eLDDs with training on projectivized data is comparable for MaltParser and MSTParser. The best strategy both overall and for eLDDs seems to be to use pseudo-projectivization with MSTParser and projective decoding, in order to use higher order factors.

6 Related Work

The Penn Treebank's annotation style of all types of long distance dependencies considerably eases the extraction of wide coverage grammars, and the building of high performing deep syntax parsers, as long as their underlying linguistic framework is able to cope with those complex phenomena (Hockenmaier et al. [9] for CCG; Cahill et al, [2] for LFG based parsing; Miyao et al. [12] for HPSG). Dependency parsers, when trained on a non projective dependency version of the PTB, and with the use of post-parsing heuristics, exhibit a similar range of performance than the parsers cited above (Nivre et al. [16]). For French, as noted in introduction, non local dependencies were not natively annotated in the FTB. This complicates of-course direct comparison of our work. Schluter and van Genabith [19] focus on

[15] We report results with the 'head' marking strategy of Nivre and Nilsson, performing very slightly better than the other two strategies (the difference not being significant).

producing treebank based LFG approximations for French, using for this a modified version of part of the FTB, where they annotated functional paths for a few cases. Clergerie [7] proposes a deep symbolic parser for French, which can recover LDDs directly (including eLDDs), but currently quantitative evaluation of LDDs for this parser remains to be performed. More generally, works on LDDs do not focus specifically on eLDDs, though they are the most difficult type of LDDs. For example, Rimell et al. [17] describe a 800 English sentences corpus used for the evaluation of wide coverage parsers on unbounded dependencies, but over all eight types of unbounded dependencies only one is exclusively of eLDD type.

7 Conclusion

We have annotated cases of *effectively long* distance dependencies for two French treebanks, totalizing over 15000 sentences. We could verify that non local dependencies are very rare in actual French texts: we annotated a non local governor for 513 tokens only out of over 420000 tokens. Yet, they are massive within the occurrences of the relative pronoun *dont*, and to a lesser extent *que*, and also frequent for the clitic *en*. We noted that extraction out of finite verbal clause is totally absent from the corpora we've annotated (one case only for over 15000 sentences), but extraction out of infinitival clause accounts for one third of the occurrences of the relative pronoun *que*. Further only 2 thirds of annotated eLDDs give rise to non projective links. As far as parsing is concerned, the best results, both overall and on eLDDs, are obtained when combining pseudo-projectivization and MSTParser with order 2 factors and projective decoding.

References

[1] Anne Abeillé and Nicolas Barrier. Enriching a french treebank. In *Proc. of LREC'04*, Lisbon, Portugal, 2004.

[2] Aoife Cahill, Michael Burke, Ruth O'Donovan, Josef van Genabith, and Andy Way. Long-Distance Dependency Resolution in Automatically Acquired Wide-Coverage PCFG-Based LFG Approximations. In *Proc. of ACL'04*, pages 320–327, Barcelona, Spain, 2004.

[3] Marie Candito, Benoit Crabbé, and Pascal Denis. Statistical french dependency parsing : Treebank conversion and first results. In *Proc. of LREC'2010*, Valletta, Malta, 2010.

[4] Marie Candito, Joakim Nivre, Pascal Denis, and Enrique Henestroza Anguiano. Benchmarking of statistical dependency parsers for french. In *Proc. of COLING 2010*, pages 108–116, 2010.

[5] Marie Candito and Djamé Seddah. Le corpus sequoia : annotation syntaxique et exploitation pour l'adaptation d'analyseur par pont lexical. In *Proc. of TALN 2012 (in French)*, Grenoble, France, June 2012.

[6] Grzegorz Chrupała. Morfette: A tool for supervised learning of morphology. `http://sites.google.com/site/morfetteweb/`, 2010. Version 0.3.1.

[7] Éric De La Clergerie. Convertir des dérivations TAG en dépendances . In *Proc. of TALN 2010*, Montreal, Canada, 2010. ATALA.

[8] Danièle Godard. Extraction out of NP in French. *Natural Language and Linguistic Theory*, 10(2):233–277, 1992.

[9] Julia Hockenmaier. *Data and models for statistical parsing with Combinatory Categorial Grammar*. PhD thesis, 2003.

[10] David M. Magerman. Statistical decision-tree models for parsing. In *Proc. of ACL'95*, pages 276–283, Morristown, NJ, USA, 1995.

[11] Ryan T. McDonald and Fernando C. N. Pereira. Online learning of approximate dependency parsing algorithms. In *Proc. of EACL'06*, 2006.

[12] Yusuke Miyao and Jun'ichi Tsujii. Probabilistic disambiguation models for wide-coverage hpsg parsing. In *Proc. of ACL 2005*, pages 83–90, 2005.

[13] Thomas Morton and Jeremy LaCivita. Wordfreak: an open tool for linguistic annotation. In *Proc. of NAACL 2003, Demos*, pages 17–18, 2003.

[14] Joakim Nivre, Johan Hall, and Jens Nilsson. Maltparser: A data-driven parser-generator for dependency parsing. In *Proc. of LREC'06*, pages 2216–2219, Genova, Italy, 2006.

[15] Joakim Nivre and Jens Nilsson. Pseudo-projective dependency parsing. In *Proc. of ACL'05*, pages 99–106, Stroudsburg, PA, USA, 2005.

[16] Joakim Nivre, Laura Rimell, Ryan McDonald, and Carlos Gomez-Rodriguez. Evaluation of dependency parsers on unbounded dependencies. In *Proc. of COLING 2010*, pages 833–841, 2010.

[17] Laura Rimell, Stephen Clark, and Mark Steedman. Unbounded dependency recovery for parser evaluation. In *Proc. of EMNLP 2009*, pages 813–821, 2009.

[18] Natalie Schluter and Josef van Genabith. Preparing, restructuring, and augmenting a French Treebank: Lexicalised parsers or coherent treebanks? In *Proc. of PACLING 07*, Melbourne, Australia, 2007.

[19] Natalie Schluter and Josef Van Genabith. Dependency parsing resources for french: Converting acquired lexical functional grammar f-structure annotations and parsing f-structures directly. 2009.

Logical Form Representation for Linguistic Resources

Rodolfo Delmonte

Department of Linguistic Studies & Department of Computer Science

Ca' Foscari University of Venice

Email: delmont@unive.it

Website: http://project.cgm.unive.it

Abstract

Logical Forms are an exceptionally important linguistic representation for highly demanding semantically related tasks like Question/ Answering and Text Understanding, but their automatic production at runtime is higly error-prone. The use of a tool like XWNet and other similar resources would be beneficial for all the NLP community, but not only. The problem is: Logical Forms are useful as long as they are consistent, otherwise they would be useless if not harmful. Like any other resource that aims at providing a meaning representation, LFs require a big effort in manual checking order to reduce the number of errors to the minimum acceptable – less than 1% - from any digital resource. As will be shown in detail in the paper, XWNet suffers from lack of a careful manual checking phase, and the number of errors is too high to make the resource usable as is. We classified mistakes by their syntactic or semantic type in order to facilitate a revision of the resource that we intend to do using regular expressions. We also commented extensively on semantic issues and on the best way to represent them in Logical Forms. In particular, we will discuss in detail representations related to three-place predicates, with special reference to secondary predication.

Keywords: Language Resources, Logical Form, Treebank consistency

1 INTRODUCTION

In a number of recent papers, the need for a sizable (at least same size of WordNet) and publicly available corpus with Logical Form representation has increased: as a result more and more papers are concerned with the generation of a logical form or a semantic representation that is close to it. The fact is that there is already a number of such resources available, XWN (Moldovan and Rus, 2001), and ILF (Agerri and Peñas, 2010), hence (AP), both derived from WordNet glosses: so, why not using them. After reviewing previous work - including XWN and WN30-lfs (by Clark et al., 2008) generated by USC/ISI, California in 2006 - AP come to the conclusion that "... there is still some need for providing lexical and/or knowledge resources suitable for computational semantics tasks that required formalized knowledge." (ibid.29) The problem seems to lie in the presence of some obscurity in the way in which the glosses have been transformed: in particular, WN30-lfs is commented by the same authors as containing "... free variables and/or predicates without any relation with any other predicates in the

definition"(ibid.29). Similar problem are also present in XWN2 (ibid.,28), where in addition, the output is cluttered with elements of the gloss which do not contribute to the definition strictly speaking, that is examples coming with the gloss. Also Clark et al. complain about the lack of consistency of XWN, but in this case no details are given in the paper.

However, not all published comments on XWN speak negatively - without any detailed analysis, in fact - of XWN: all published work by the authors of XWN speaks in favour of it. There are many papers published by the authors, V.Rus, D.Moldovan, S.Harabagiu et al., R.Mihalcea et al. – see the References -, who describe their work positively, if not highly positively, and comment on its usefulness for various semantically heavy tasks like Question Answering and RTE. In particular, Rus V., 2001 indicates an experiment with evaluation, where the accuracy for glosses conversion into Logical Forms is reported at 89.46%. However, results are obtained on a selection of 1000 WN glosses only. If this level of accuracy could be applied to the whole of the resource, the final result would be an error rate slightly over 10%: this could be regarded positively. In fact, I found over 30% error rate, and this is why – in my opinion - the XWN is badly flawed and cannot be used for the purpose it was made.

I don't want to imply that work carried out is useless: on the contrary, since it can improved I intend to correct it in the future, and I started by provinding classes of mistakes which seems to me the best way to help doing that. A lot of difficult problems have been solved in XWN that deserve the resource to be saved and improved upon. Producing such a resource from scratch is outside the scope of current NLP technology, and this is attested by the various attempts at achieving such a goal (see also Ovchinnikova et al., 2011). However, there are also other attempts at producing Logical Forms directly from Penn Treebank style syntactic representations, like for instance, the LFToolkit by Nishit Rashod and Jerry Hobbs at their website, and the experiment reported by Alshawi et al. that are commented on here below.

In Alshawi et al. (2011) an experiment is reported to derive sentence-semantics pairs for training and testing from the Penn Treebank. In order to do that they program the Stanford treebank toolkit to produce what they call NLF expressions, that is Natural Logical Form, which are intentionally not intended as fully resolved logical forms. These are meant to be closer to natural logic than QLF Quasi Logical Forms, and are being built in order to use them to make some Natural Logic inference. As the authors themselves comment, QLFs are being used widely to refer to any logic-like semantic representation without explicit quantifier scope, i.e. unscoped logical forms(ibid.17). In the same paper the authors specifically comment on the need to use an unknown/unspecified Null operator, %, for all those linguistic constructs which are beyond the coverage of their semantic model. This applies to a great number of constructions that are present in the PTB. As a result of the experiment the accuracy is around 86%. Here I have to note that the usefulness of such logic-like representation is very low due to incompleteness of its final results.

The Null operator is also present in PTB for all those linguistic constructions that have been regarded too difficult to take decisions upon by annotators. These constructions typically include all adjunct infinitivals and

gerundives for a total amount of some 12,000 non coindexed null elements over some 38,000 Null Elements. This problem has also prevented other attempts at producing a semantically viable corpus of logical forms directly from a mapping of PTB, by a number of other researchers working in the LFG framework, (Guo et al., 2007) and in HPSG and CCG frameworks, but also Dependency Grammar as reported in (Nivre and Nilsson, 2005).

In Branco 2009, the author reviews a possible annotation process for a yet to be constructed resource, which is correctly regarded, the "next generation of semantically annotated corpora" (ibid.6). However, since the author does not make any reference to real existing resources, the whole discussion remains very theoretical. In a subsequent paper (Branco et al. 2012), the same author presents a parser for the construction of what he calls "deep linguistic databank, called CINTIL DeepGramBamk" (ibid, 1810). In fact, the authors depict the process of creating a Logical Form as a side effect,

> "As a side effect, it permits to obtain very important payoffs: as the deep linguistic representation of a sentence may encode as much grammatical information as it is viable to associate to a sentence, by constructing a deep linguistic databank one is producing in tandem, and within the same amount of effort, a POS-tagged corpus, a constituency TreeBank, a DependencyBank, a PropBank, or even a LogicalFormBank."

This is clearly an underestimation of the real problem that has to be solved when moving from a constituency structure-based representation to other levels of representation, where additional information needs to be added, as we will discuss below. In the two papers by Branco quoted above, the authors never refer to existing Logical Form resources, as if there was no other effort in that direction done and accomplished by others.

All these methods go beyond the encoding of surface context-free phrase structure trees, to incorporate non-local dependencies. This option requires recovering empty nodes and identifying their antecedents, be they traces or long distance dependencies. But since PTB annotators themselves intentionally refused to coindex all those cases that caused some difficulty in the decision process, all work carried out on this resource is flawed, semantically speaking, from the start. However, I must admit to the fact that WN glosses are much simpler sentences in comparison to PTB sentences, which even if taken with a word limit under 40 are still too complex and not comparable to definitions.

In a previous paper(Delmonte & Rotondi, 2012) I revised the typical mistakes present in the corpus and commented on them; I also compared XWN with the representation contained in other similar resources. In this paper I will limit myself to XWN and I will extend the previous analysis. In particular, in section 2 below I will introduce and comment at length the thorny problem of representing three-place predicates in LF. Then I will add some conclusion.

2 The Problem of Three-Place Predicates and Their Representation in LF

Logical Forms in XWN are graded in three quality levels: normal, silver and gold; the same applies to tagging and phrase structure constituency. "Normal"

quality, which applies to the majority of the glosses, is used to indicate that there is no agreement between the two parsers that have been used to parse the input definition, and that there has been no manual checking of the output. "Gold" quality means manual checking has been performed, and "silver" quality indicates that there has been no manual checking but the two parsers agree in their representation. The importance given to the agreement between the two constituency parsers, is due to the fact that LFs are a mapping on syntactic constituency representation.

LF from glosses is represented in different manner according to lexical category, adjective, verb, noun and adverb: each one is associated to a predicate but with some differences. As far as verbs are concerned we have the following picture:
For each synset, a variable 'e1' is associated to the first term that represents it, to indicate the eventuality of the action/state/event of the verb meaning; the subject is associated invariably to 'x1' and the object to 'x2'. The second argument may be fictitious in case of intransitive verbs.

recognize:VB(e1, x1, x2) -> show:VB(e1, x1, x5) approval:NN(x3) or:CC(x5, x3, x4) appreciation:NN(x4) of:IN(x5, x2)

In this case all variables are bound to some argument position and are associated to some linguistic element.
In the case of ditransitive verbs the LF representation of the event is verb(e1,x1,x2,x3), as in, *professor gives students the grades: professor(x1) give(e1, x1, x2, x3) grade(x2) student (x3),* however this representation is not the rule. This issue constitutes by itself a thorny problem even at a theoretical level (but see Pustejovsky 1991;2000), so I will start by introducing the topic and then I will delve into the way in which it has been treated in XWN. There are at least four different types of three-place predicates (hence 3PPs):

a. PUT, John put the book on the table
b. STRIP, John strips capital gains from the bill
c. CONSIDER, John considers Mary silly
d. CAUSE, John caused the car to stop

These examples could be multiplied with other logically and lexically similar verbs and I will add one more example in the logical forms below. One also needs to stress the fact that example c. is a case of secondary predication which is semantically identical to "John painted the house black", but lexically totally different. The verb PAINT is just a transitive verb, which can at times receive secondary predication, though necessarily semantically viable, like a colour "BLACK". The verb CONSIDER is lexically speaking, a 3-place predicate, i.e. a special type of transitive verb. Special treatment is required also for "perception verbs" like SEE and others that we discuss in detail below. What is important now, is to highlight the lexical structures of each case: Case A. has a Subject, an Object and an Oblique which is an Oblique complement and is Obligatory. Differently from Case B. where the verb STRIP has an Oblique which is an Optional argument. Case A. could also be exemplified by verbs like GIVE, TELL, PROMISE etc. – as for instance in “John gave Mary a book” - which can exhibit a double object construction in addition to the

object+indirect object/object one. Logical forms for these two cases are below:

- PUT(e1,x1,x2,x3), John(x1),book(x2),table(x3)
- GIVE(e1,x1,x2,x3), John(x1),Mary(x3),book(x2)
- STRIP(e1,x1,x2), John(x1),gain(x2),capital(x2), bill(x3) from(e1,x3).

where the BILL is treated as Optional or as an Adjunct. The remaining two cases are frequently made a topic of discussion in the theoretical literature because they are linked to the question of Small Clauses (SC), a thorny issue to represent in logical form. Small clauses are presented in the already quoted PennTreebank guide, where they are commented as follows:

> "The bare infinitive complements of perception verbs (see, hear, feel) and causative verbs (make, let, have; also help) are bracketed together with the NP preceding them as a complement S. The structural subjects of both the matrix clause and the embedded clause are tagged -SBJ.
>
> (S (NP-SBJ I)
> (VP saw
> (S (NP-SBJ him)
> (VP do
> (NP it)))))
>
> Adjectival predicates introduced by as are labeled PP-CLR, with no small clause.
>
> (S (NP-SBJ They)
> (VP do not
> (VP consider
> (NP (NP themselves)
> and
> (NP their plight))
> (PP-CLR as
> (ADJP statistical)))))
>
> Nominal predicates Verbs with two NP complements receive either a small clause analysis:
>
> (S (NP-SBJ The late
> (NAC Secretary (PP of (NP State)))
> John Foster Dulles)
> (VP considered
> (S (NP-SBJ the 1954 Geneva agreement)
> (NP-PRD (NP a specimen)
> (PP of (NP appeasement))))))
>
> or a double object analysis:
>
> (S (NP-SBJ His bel canto style)
> (VP gave
> (NP the performance)
> (NP a special distinction)))
>
> …Verbs that can take this kind of Small Clause include hold, keep, leave, call, pronounce; wish; believe, consider, find, imagine, think;

appoint, elect, make, vote; certify, christen, declare, name, among others."(ibid.97;107).

Generalizing a small clause analysis in constituent structure is certainly useful, but we must also stress the fact that the theory underlying the construction of small clauses is not adequate for their treatment in LFs. Blurring the difference between the first pair of verbs I already introduced above, and the second pair, is harmful in many ways. In particular, in the case of of PUT and GIVE what we want to say is basically that,

"there is a PUTting/GIVing event of the BOOK and the event is ***completed*** by the change_of_possession/change_of_position to a GOAL/RECIPIENT/LOCATION argument"

The need ***to complete the meaning*** by means of the third argument is what makes the difference with the case of Optional Arguments. The same argument may be applied to verbs like CONSIDER CAUSE and SEE that we exemplify now with their LFs:

-CONSIDER(e1,x1,x2,e2),John(x1),Mary(x2),silly(e2,x2)
-CAUSE(e1,x1,x2,e2), John(x1), car(x2), stop(e2,x1,x2)

In these LFs it is the ***secondary predication that completes the meaning*** with its own event structure. The secondary predication introduces a new event variable which is used to associate its meaning to the appropriate argument: it is Mary that is SILLY and not John; it is the car that STOPs and not John. Besides, both the CAR and MARY are real OBJect arguments of the main verb, which can be passivized:
- Mary is considered silly (by many people).
- The car was stopped (by a policeman).
In other words, the Small Clause analysis treats the OBJect of the matrix clause as SUBJect of the SC in constituent structure, and this is basically wrong. What I am interested in, is checking the way in which these constituent structures have been represented in XWN and how the SC analysis has influenced the LF mapping. I will start by showing two examples of the way in which PUT is encoded, wrongly: the verb CUP, and the verb PATCH.

put into a cup; "cup the milk"
(S (S (VP (TO to)
(VP (VB cup))))
(VP (VBZ is)
(S (VP (TO to)
(VP (VB put)
(PP (IN into)
(NP (DT a) (NN cup)))))))
cup:VB(e1, x1, x2) -> put:VB(e1, x1, x4) into:IN(e1, x3) cup:NN(x3)

mend by putting a patch on; "patch a hole"
(S (S (VP (TO to)

```
          (VP (VB patch) ) ) )
      (VP (VBZ is)
        (S (VP (TO to)
             (VP (VB mend)
               (PP (IN by)
                 (S (VP (VBG putting)
                      (NP (DT a) (NN patch) )
                      (PP (IN on) ) ) ) ) ) )
patch:VB(e1, x1, x2) -> mend:VB(e1, x1, x2) by:IN(e1, e3) put:VB(e3, x1, x5)
patch:NN(x5) on:IN(e3, x2)
```

In both cases, the verb PUT is encoded without the OBLique, which is then added as an ADJunct by means of the semantics of the preposition IN. In addition, the verb CUP does not tranfer the OBJect argument to the verb PUT which is left without arguments apart from the SUBJect x1; both x2 and x4 are treated as free variables. In the whole database, there are only three cases where the verb PUT is correctly encoded as a 3PP, and one of them is here below:

```
<synonymSet>make, make_up</synonymSet>
 <text> put in order or neaten; "make the bed"; "make up a room"    </text>
<lft quality="GOLD">
make:VB(e1, x1, x2) -> put:VB(e2, x1, x3, x2) in_order:JJ(x3) or:CC(e1, e2, e3)
neaten:VB(e3, x1, x2)

<synonymSet>caution, admonish, monish</synonymSet>
 <text> warn strongly; put on guard  </text>
<lft quality="GOLD">
caution:VB(e1, x1, x2) -> put:VB(e1, x1, x3, x2) on_guard:JJ(x3)
```

The fact that the third argument is computed as an ADJectival and not as a Prepositional ADJunct may explain its peculiar treatment, which is only found in similar contexts. In fact this treatment has also been used for MAKE when it is used with secondary predications as shown in the following two entries:

```
<synonymSet>darkened</synonymSet>
   <text>    become or made dark  </text>
darkened:JJ(x1) -> make:VB(e2, x5, x4, x1) dark:JJ(x4)

<synonymSet>intimidated</synonymSet>
  <text> made timid or fearful as by threats   </text>
intimidated:JJ(x1) -> make:VB(e1, x6, x4, x1) timid:JJ(x4) fearful:JJ(x4) as_by:IN(e1,
x2) threat:NN(x2)
```

Here correctly, we see that MAKE is encoded as a 3PP and all variables as bound, apart from the SUBJect which is left implicit in all entries of this type.
Finally I will look at CAUSE, which should be given a structure similar to FORCE, at least whenever an infinitival occurs as complement. Some examples here below:

<synonymSet>haemagglutinate, hemagglutinate</synonymSet>
<text> cause the clumping together </text>
(VP (TO to)
(VP (VB cause)
(NP (DT the)
(ADJP (VBG clumping) (RB together)))
haemagglutinate:VB(e1, x1, x2) -> cause:VB(e1, x1, e2) clump:VB(e2, x4, x2) together:RB(e2)

<synonymSet>set_ablaze, set_aflame, set_on_fire, set_afire</synonymSet>
<text> set fire to; cause to start burning </text>
(VP (TO to)
(VP (VB cause)
(S (VP (TO to)
(VP (VB start)
(S (VP (VBG burning)))
set_ablaze:VB(e1, x1, x2) -> cause:VB(e1, x1, x2) to:IN(e1, e2) start:VB(e2, x2, e3) burn:VB(e3, x2, x3)

<synonymSet>shame</synonymSet>
<text> cause to be ashamed </text>
(VP (VB cause)
(S (VP (TO to)
(VP (VB be)
(ADJP (JJ ashamed)))
shame:VB(e1, x1, x2) -> cause:VB(e1, x1, x4) to:IN(e1, e2) be:VB(e2, x4, x3) ashamed:JJ(x3)

Apart from the first example, HAEMAGGLUTINATE, which is correctly encoded, the two other cases are totally wrong. In particular, SHAME, contains a free variable X2, which is at first substituted by X4 and then by X3. In the tables here below I analyse the data related to the encoding of MAKE and CAUSE in LFs. As shown in Table 2., the use of MAKE in the 3PP format constitutes approximately 12% (457) of the total 3351 occurrences, where the rest on the contrary encodes MAKE basically as a two-place predicate. The other partly similar predicate, CONSIDER, is shown in Table 3. and has a different behaviour: it is used as a 3PP 8% (28) of the total occurencies (369), a very small amount if compared to MAKE, but still notable when compared to the synonym verb REGARD, which appears 120 times, always encoded as two-place predicate and never encoded as 3PP. Finally, CAUSE appears only 35 times correctly encoded as a two-place predicate with another eventive argument encoding the SC. The remaining cases are wrong two-place encoding.

Types	**Adverb.**	**Adject.**	**Verbs**	**Nouns**	**Total**
VB(e1, x1, x4)	4	244	537	714	1499

Types	Adverb.	Adject.	Verbs	Nouns	Total
VB(e1, x1, e2)	0	5	2	28	35
VB(e1, x1)	0	8	1	0	9
VB(e1,x1, x2, x4)	0	0	0	4	4
Total	4	257	540	746	1547

Table 1.: Number of different types of eventive structures for CAUSE in LF entries

Types	Adverb.	Adject.	Verbs	Nouns	Total
VB(e1, x1, x4)	5	193	846	1751	2795
VB(e1, x1, e2)	0	3	21	56	80
VB(e1, x1)	0	12	7	0	19
VB(e1,x1, x2, x4)	1	112	307	37	457
Total	6	320	1181	1844	3351

Table 2.: Number of different types of eventive structures for MAKE in LF entries

Types	Adverb.	Adject.	Verbs	Nouns	Total
VB(e1, x1, x4)	0	14	52	270	336
VB(e1, x1, e2)	0	0	0	5	5
VB(e1, x1)	0	0	0	0	0
VB(e1,x1, x2, x4)	0	3	3	22	28
Total	0	17	55	297	369

Table 3.: Number of different types of eventive structures for CONSIDER in LF entries

4 Some general consideration on XWN

Some general considerations over the whole dataset come from considering the amount of GOLD data with respect to NORMAL or SILVER, as shown in Table 4 below.

Types	Adverb.	Adjectiv.	Verbs	Nouns
Gold	3994	16059	14441	32844
Silver	0	4321	0	7228
Normal	0	0	0	54796

Types	Adverb.	Adjectiv.	Verbs	Nouns
Total	3994	20380	14441	94868

Table 4.: Number of Gold/Silver/Normal LF entries in XWN

As can be easily gathered, the number of errors will vary substantially from one file to the other depending strictly on the number of GOLD LF entries, and will be proportional to the overall size of the file in terms of total number of entries. The file in which most errors are found is the one of NOUNS, which is not only the only file to contain Normal entries, but also in a quantity which is much higher than the GOLD ones, almost the double. Another important factor that may be considered as possible cause of errors in the NOUN file is the length of the gloss in number of words, which is more extended in syntactic terms than in the other files.

As a final remark, we extracted all the records containing just the LF from every single file, we then sorted them and checked for their consistency: this was done in order to verify that no two Logical Forms are identical to each other. Whenever this happens, the meaning associated to one synset would be interchangeable with the meaning associated to another synset, which is clear sign of inconsistency. We found the following situation,

- over 94868 entries for Nouns, 43 are duplicate LFs
- over 20380 entries for Adjective, 47 are duplicate LFs
- over 3994 entries for Adverbs, 12 are duplicate LFs
- over 14441 entries for Verbs, 29 are duplicatre LFs

Here below we report some examples of duplicate, or sometimes triple LF representations taken from the Noun file:

alaska_peninsula:NN(x1) -> peninsula:NN(x1) in:IN(x1, x2) southwestern:JJ(x2) alaska:NN(x2)

alpaca:NN(x1) -> wool:NN(x1) of:IN(x1, x2) alpaca:NN(x2)

anagoge:NN(x1) -> mystical:JJ(x1) allegorical:JJ(x1) interpretation:NN(x1)

approbation:NN(x1) -> official:JJ(x1) approval:NN(x1)

bailey:NN(x1) -> outer:JJ(x1) courtyard:NN(x1) of:IN(x1, x2) castle:NN(x2)

Bernoulli:NN(x1) -> swiss:JJ(x1) mathematician:NN(x1)

blood_count:NN(x1) -> number:NN(x1) of:IN(x1, x2) red:JJ(x2) white:JJ(x2) corpuscle:NN(x2) in:IN(x2, x3) blood:NN(x3) sample:NN(x4)

card_catalog:NN(x1) -> enumeration:NN(x1) of:IN(x1, x2) resource:NN(x2) of:IN(x2, x3) library:NN(x3)

cassava:NN(x1) -> source:NN(x1) of:IN(x1, x2) tapioca:NN(x2)

catapult:NN(x1) -> use:VB(e1, x2, x1) to:IN(e1, e2) propel:VB(e2, x1, x1) small:JJ(x1) stone:NN(x1)

clash:NN(x1) -> state:NN(x1) of:IN(x1, x2) conflict:NN(x2) between:IN(x2, x3) person:NN(x3)

5 Conclusion

Eventually we may comment that there are a number of resources available with Logical Forms representations of WordNet glosses, and a number of algorithms which can be used off-the-shelf to produce Logical Forms from PTB constituency based phrase structure representations: none of these resources is however usable as is, do to error rates which average 30%. Improvements can be achieved by manual correction of all the LFs contained in these resources. This is an option that we intend to carry out in a local project that will be the followup of a MA degree thesis that started this research. Work to accomplish this task requires annotators which have been previously trained and have acquired the needed skill to read logical forms. We think a skilled annotator will need 6 month work to correct all types of mistakes we found.

The research has focussed on the typing of the mistakes present in the resource itself: this has been made easier by the fact that in both resources analysed, the conversion into LFs has started from the output of a syntactic parser – in the case of XWN, two constituency parsers, while in ILF, one dependency parser. The result of the manual corrections will be made available online to be accessed freely by anyone interested in using them.

References

Agerri, R. and Anselmo Peñas (2010). *On the Automatic Generation of Intermediate Logic Form for WordNet glosses*, 11th International Conference on Intelligent Text Processing and Computational Linguistics (Cicling-2010), LNCS Vol. 6008, Springer.

Alshawi, H., Pi-Chuan Chang, M. Ringgaard. (2011). Deterministic Statistical Mapping of Sentences to Underspecified Semantics, in Johan Bos and Stephen Pulman (editors), Proceedings of the 9th International Conference on Computational Semantics, IWCS,15-24.

Branco, A. 2009, "LogicalFormBanks, the Next Generation of Semantically Annotated Corpora: key issues in construction methodology", In Klopotek, et al., (eds.), Recent Adavnces in Intelligent Information Systems, Warsaw, 3-11.

Branco, A., F. Costa, J. Silva, S. Silveira, S. Castro, M. Avelãs, C. Pinto and J. Graça, 2010, "Developing a Deep Linguistic Databank Supporting a Collection of Treebanks: the CINTIL DeepGramBank ", In Proceedings, LREC2010, La Valleta, Malta, 1810-1815.

Clark, P., Fellbaum, C., Hobbs, J.R., Harrison, P., Murray, W.R., Thompson, J.: Augmenting WordNet for Deep Understanding of Text. In: Bos, J., Delmonte, R. (eds.) Semantics in Text Processing. STEP 2008 Conference Proceedings. Research in Computational Semantics, vol. 1, pp. 45–57. College Publications (2008)

Delmonte R., (2007). Computational Linguistic Text Processing – Logical Form, Semantic Interpretation, Discourse Relations and Question Answering, Nova Science Publishers, New York.

Delmonte, R., A.Rotondi, 2012. "Treebanks of Logical Forms: they are Useful Only if Consistent", in Proceedings of the Workshop SAIIMRT LREC, Istanbul, www.lrec-conf.org/proceedings/LREC2012/Workshops/19.SAIIMRT /Proceedings.pdf, 9-16.

Yuqing Guo, HaifengWang, Josef van Genabith, Recovering Non-Local Dependencies for Chinese, 2007. Proc. of the Joint Conference on EMNLP and

Computational Natural Language Learning, 257–266.
Harabagiu, S.M., Miller, G.A., Moldovan, D.I.: eXtended WordNet - A Morphologically and Semantically Enhanced Resource (2003), http://xwn.hlt.utdallas.edu, 1-8.
Information Science Institute, University of Southern California: Logical Forms for WordNet 3.0 glosses (2007), http://wordnetcode.princeton.edu/standoff-files/wn30-lfs.zip
Mihalcea, R., and Dan I. Moldovan, (2001). *eXtended WordNet: progress report,* In: Proceedings of NAACL Workshop on WordNet and Other Lexical Resources, Pittsburgh, 95-100.
Moldovan, D., Rus, V.: Explaining Answers with Extended WordNet. In: Proceedings of the Association for Computational Linguistics, ACL 2001 (2001)
Moldovan, D., S. Harabagiu, R. Girju, P. Morarescu, F. Lacatusu, A. Novischi, A. Badulescu, O. Bolohan, *LCC Tools for Question Answering,* In: Proceedings of TREC, 2002.
Moldovan, D., Vasile Rus: Transformation of WordNet Glosses into Logic Forms. FLAIRS Conference 2001: 459-463.
Moldovan, D., Vasile Rus: Logic Form Transformation of WordNet and its Applicability to Question Answering. ACL 2001: 394-401.
Ovchinnikova, E., N. Montazeri, T. Alexandrov, J. R. Hobbs, M. C. McCord, R. Mulkar-Mehta, (2011). Abductive Reasoning with a Large Knowledge Base for Discourse Processing, in J. Bos and S. Pulman (editors), Proc. of the 9th International Conference on Computational Semantics, IWCS, 225-234.
Pustejovsky, J. 2000. Events and the Semantics of Opposition. In C. Tenny and J. Pustejovsky
(eds.). Events as Grammatical Objects. Stanford. CSLI Publications, pp. 445–482. Pustejovsky, J. 1995. The Generative Lexicon. Cambridge, MA. MIT Press.
Pustejovsky, J. 1991. The syntax of event structure. Cognition 41, pp. 47-81.
Ramsay, A.M. and D.G.Field (2008). Speech acts, epistemic planning and Grice's maxim. Journal of Logic and Computation 18(3), 431-457.
Rus, V. and Dan I. Moldovan: High Performance Logic Form Transformation. International Journal on Artificial Intelligence Tools 11(3): 437-454 (2002)
Rus, V., Dan I. Moldovan, Orest Bolohan, (2002). Bracketing Compound Nouns for Logic Form Derivation. FLAIRS Conference 2002: 198-202.
Rus, V., Alex Fit-Florea, (2004). A Model for Identifying the Underlying Logical Structure of Natural Language. PRICAI 2004: 977-978.
Rus, V., (2004). Experiments with Machine Learning for Logic Arguments Identification. MAICS 2004: 40-47.
Rus, Vasile, (2004). *A First Evaluation of Logic Form Identification Systems,* SENSEVAL-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text.
Vasile Rus, *High Precision Logic Form Transformation,* 2001. In Proceedings of ICTAI'2001. pp.288-288.
WordNet Gloss Disambiguation Project, Princeton University: Semantically annotated gloss corpus (2008), http://wordnet.princeton.edu/ glosstag.shtml

DeepBank: A Dynamically Annotated Treebank of the Wall Street Journal

Dan Flickinger[†], Yi Zhang[‡♯], Valia Kordoni[◇♯]
† CSLI, Stanford University
‡ Department of Computational Linguistics, Saarland University, Germany
♯ LT-Lab, German Research Center for Artificial Intelligence
◇ Department of English Studies, Humboldt University of Berlin

E-mail: danf@stanford.edu, yizhang@dfki.de,
kordonie@anglistik.hu-berlin.de

Abstract

This paper describes a large on-going effort, nearing completion, which aims to annotate the text of all of the 25 Wall Street Journal sections included in the Penn Treebank, using a hand-written broad-coverage grammar of English, manual disambiguation, and a PCFG approximation for the sentences not yet successfully analyzed by the grammar. These grammar-based annotations are linguistically rich, including both fine-grained syntactic structures grounded in the Head-driven Phrase Structure Grammar framework, as well as logically sound semantic representations expressed in Minimal Recursion Semantics. The linguistic depth of these annotations on a large and familiar corpus should enable a variety of NLP-related tasks, including more direct comparison of grammars and parsers across frameworks, identification of sentences exhibiting linguistically interesting phenomena, and training of more accurate robust parsers and parse-ranking models that will also perform well on texts in other domains.

1 Introduction

This paper presents the English DeepBank, an on-going project whose aim is to produce rich syntactic and semantic annotations for the 25 Wall Street Journal (WSJ) sections included in the Penn Treebank (PTB: [16]). The annotations are for the most part produced by manual disambiguation of parses licensed by the English Resource Grammar (ERG: [10]), which is a hand-written, broad-coverage grammar for English in the framework of Head-driven Phrase Structure Grammar (HPSG: [19]).

Large-scale full syntactic annotation has for quite some time been approached with mixed feelings by researchers. On the one hand, detailed syntactic annotation can serve as a basis for corpus-linguistic study and improved data-driven NLP

methods. When combined with supervised machine learning methods, such richly annotated language resources including treebanks play a key role in modern computational linguistics. The availability of large-scale treebanks in recent years has contributed to the blossoming of data-driven approaches to robust and practical parsing.

On the other hand, the creation of detailed and consistent syntactic annotations on a large scale turns out to be a challenging task.[1] From the choice of the appropriate linguistic framework and the design of the annotation scheme to the choice of the text source and the working protocols on the synchronization of the parallel development, as well as quality assurance, each of the steps in the entire annotation procedure presents non-trivial challenges that can impede the successful production of such resources.

The aim of the DeepBank project is to overcome some of the limitations and shortcomings which are inherent in manual corpus annotation efforts, such as the German Negra/Tiger Treebank ([2]), the Prague Dependency Treebank ([11]), and the TüBa-D/Z.[2] All of these have stimulated research in various sub-fields of computational linguistics where corpus-based empirical methods are used, but at a high cost of development and with limits on the level of detail in the syntactic and semantic annotations that can be consistently sustained. The central difference in the DeepBank approach is to adopt the *dynamic* treebanking methodology of Redwoods [18], which uses a grammar to produce full candidate analyses, and has human annotators disambiguate to identify and record the correct analyses, with the disambiguation choices recorded at the granularity of constituent words and phrases. This localized disambiguation enables the treebank annotations to be repeatedly refined by making corrections and improvements to the grammar, with the changes then projected throughout the treebank by reparsing the corpus and re-applying the disambiguation choices, with a relatively small number of new disambiguation choices left for manual disambiguation.

For the English DeepBank annotation task, we make extensive use of resources in the DELPH-IN repository[3], including the PET unification-based parser ([4]), the ERG plus a regular-expression preprocessor ([1]), the LKB grammar development platform ([7]), and the [incr tsdb()] competence and performance profiling system ([17]), which includes the treebanking tools used for disambiguation and inspection. Using these resources, the task of treebank construction shifts from a labor-intensive task of drawing trees from scratch to a more intelligence-demanding task of choosing among candidate analyses to either arrive at the desired analysis or reject all candidates as ill-formed. The DeepBank approach should be differentiated from so-called treebank conversion approaches, which derive a new treebank

[1] Besides[18], which we draw more on for the remainder of the paper, similar work has been done in the HPSG framework for Dutch [22]. Moreover, there is quite a lot of related research in the LFG community, e.g., in the context of the ParGram project: [9] for German, [14] for English, and the (related) Trepil project, e.g., [20] for Norwegian.

[2] http://www.sfs.nphil.uni-tuebingen.de/en_tuebadz.shtml

[3] http://www.delph-in.net

from another already existing one, such as the Penn Treebank, mapping from one format to another, and often from one linguistic framework to another, adapting and often enriching the annotations semi-automatically. In contrast, the English DeepBank resource is constructed by taking as input only the original 'raw' WSJ text, sentence-segmented to align with the segmentation in the PTB for ease of comparison, but making no reference to any of the PTB annotations, so that we maintain a fully independent annotation pipeline, important for later evaluation of the quality of our annotations over held-out sections.

2 DeepBank

The process of DeepBank annotation of the Wall Street Journal corpus is organised into iterations of a cycle of parsing, treebanking, error analysis and grammar/treebank updates, with the goal of maximizing the accuracy of annotation through successive refinement.

Parsing Each section of the WSJ corpus is first parsed with the PET parser using the ERG, with lexical entries for unknown words added on the fly based on a conventional part-of-speech tagger, TnT [3]. Analyses are ranked using a maximum-entropy model built using the TADM [15] package, originally trained on out-of-domain treebanked data, and later improved in accuracy for this task by including a portion of the emerging DeepBank itself for training data. A maximum of 500 highest-ranking analyses are recorded for each sentence, with this limit motivated both by practical constraints on data storage costs for each parse forest and by the processing capacity of the [incr tsdb()] treebanking tool. The existing parse-ranking model has proven to be accurate enough to ensure that the desired analysis is almost always in these top 500 readings if it is licensed by the grammar at all. For each analysis in each parse forest, we record the exact derivation tree, which identifies the specific lexical entries and the lexical and syntactic rules applied to license that analysis, comprising a complete 'recipe' sufficient to reconstruct the full feature structure given the relevant version of the grammar. This approach enables relatively efficient storage of each parse forest without any loss of detail.

Treebanking For each sentence of the corpus, the parsing results are then manually disambiguated by the human annotators, using the [incr tsdb()] treebanking tool which presents the annotator with a set of binary decisions, called *discriminants*, on the inclusion or exclusion of candidate lexical or phrasal elements for the desired analysis. This discriminant-based approach of [6] enables rapid reduction of the parse forest to either the single desired analysis, or to rejection of the whole forest for sentences where the grammar has failed to propose a viable analysis.[4] On average, given n candidate trees, $\log_2 n$ decisions are needed in order to

[4]For some sentences, an annotator may be unsure about the correctness of the best available analysis, in which case the analysis can still be recorded in the treebank, but with a lower 'confidence'

fully disambiguate the parse forest for a sentence. Given that we set a limit of 500 candidate readings per sentence, full disambiguation of a newly parsed sentence averages no more than 9 decisions, which enables a careful annotator to sustain a treebanking rate of 30 to 50 sentences per hour on the first pass through the corpus.

Error analysis During the course of this annotation effort, several annotators have been trained and assigned to carry out the initial treebanking of portions of the WSJ corpus, with most sections singly annotated. On successive passes through the treebank, two types of errors are identified and dealt with: mistakes or inconsistencies of annotation, and shortcomings of the grammar such that the desired analysis for a given sentence was not yet available in the parse forest. Errors in annotation include mistakes in constituent boundaries, in lexical choice such as verb valency or even basic part of speech, and in phrasal structures such as the level of attachment of modifiers or the grouping of conjuncts in a coordinated phrase. Our calculation of the inter-annotator agreeemnt using the Cohen's KAPPA[5] on the constituents of the derivation trees after the initial round of treebanking shows a moderate agreement level at $\kappa = 0.6$. Such disagreeements are identified for correction both by systematic review of the recorded 'correct' trees section by section, and by searching through the treebank for specific identifiers of constructions or lexical entries known to be relatively rare in the WSJ, such as the rules admitting questions or imperative clauses.

Shortcomings of the grammar are identified by examining sentences for which annotators did not record a correct analysis, either because no analysis was assigned, or because all of the top 500 candidate analyses were flawed. Some of the sources of error emerge quickly from even cursory analysis, such as the initial absence of a correct treatment in the ERG for measure phrases used as verbal modifiers, which are frequent in the WSJ corpus, as in *the index rose 20 points* or *the market fell 14%*. Other types of errors required more detailed analysis, such as missing lexical entries for some nouns taking verbal complements, as in *the news that Smith was hired* or *the temptation to spend the money*. These fine-grained lexical entries are not correctly predicted on the fly using the part-of-speech tagger, and hence must be added to the 35,000-entry manually supplied lexicon in the ERG.

Grammar & Treebank Update While grammar development proceeds independent of the initial treebank annotation process, we have periodically incorporated improvements to the grammar into the treebank annotation cycle. When a grammar update is incorporated, the treebank also gets updated accordingly by (i) parsing anew all of the sentences in the corpus using the new grammar; (ii) reapplying the recorded annotation decisions; and (iii) annotating those sentences which are not fully disambiguated after step ii, either because new ambiguity was introduced by the grammar changes, or because a sentence which previously failed

score assigned, so the annotation can be reviewed in a later cycle of updates.

to parse now does. The extra manual annotation effort in treebank update is relatively small when compared to the first round of annotation, typically requiring one or two additional decisions for some 5–10% of the previously recorded correct analyses, and new annotation for previously rejected items, which were another 15% of the total in the second round, and much less in successive rounds. Hence these later rounds of updating the treebank proceed more quickly than the initial round of annotation.

Correcting errors of both classes based on analysis of the first pass through DeepBank annotation has resulted in a significant improvement in coverage and accuracy for the ERG over the WSJ corpus. Raw coverage has risen by some 10% from the first pass and the 'survival' rate of successfully treebanked sentences has risen even more dramatically to more than 80% of all sentences in the first 16 sections of the WSJ that have now gone through two rounds of grammar/treebank updates. The table below shows the current status of these first 16 sections of the English DeepBank in terms of "Observed" and "Verified" coverage, where the former reports the number of sentences that received at least one analysis from the ERG, and the latter gives the number of sentences for which the annotator recorded a correct analysis.

Table 1: English DeepBank ERG results for WSJ Sections 00–15

Section	Number of items	Observed coverage	Verified coverage
00	1922	92.2%	82.0%
01	1997	92.3%	81.6%
02	1996	92.3%	84.0%
03	1482	92.0%	82.1%
04	2269	92.6%	81.5%
05	2137	92.3%	81.8%
06	1835	91.3%	81.1%
07	2166	91.9%	82.6%
08	478	90.6%	80.1%
09	2073	92.0%	81.2%
10	1945	91.8%	81.3%
11	2237	91.5%	80.4%
12	2124	94.2%	85.1%
13	2481	94.8%	85.8%
14	2182	94.0%	86.0%
15	2118	94.1%	86.4%
Subtotal	31442	92.6%	82.6%

These figures, which are surprisingly stable across sections both in raw parsing coverage and in treebanked items, show that roughly 18% of the sentences in the corpus fail to receive a correct analysis from the ERG; we discuss the DeepBank annotations for this portion of the corpus in section 4. Note that most of the remaining portion of the WSJ corpus has now been treebanked the first time through, and we expect the remaining updated sections to be completed by the end of the year, excluding three held-out sections reserved for future testing.

3 Annotation formats

In comparison to existing large-scale treebanks, DeepBank stands out as a unique resource which incorporates both syntactic and semantic annotations in a uniform grammar framework. To facilitate the easy access of various layers of annotation in the treebank, multiple formats will be provided in the release of the English DeepBank. The [incr tsdb()] profiles are comprehensive relational databases that record the original ERG derivation trees together with the semantic representations natively expressed in Minimal Recursion Semantics (MRS: [8]) structures. The database also keeps the history of the manual annotations (the disambiguation discriminants). For users interested in simpler or more conventional representations, the HPSG derivations are also converted to PTB-style phrase structure tree representations which employ a mapping of HPSG categories to a smaller set of POS and phrasal categories that roughly corresponding to those of the English PTB. Furthermore, the treebank is also available in a dependency-oriented representation following the format of the CoNLL-2008 Shared Task [21]. The syntactic dependencies are extracted from the derivation trees of the ERG, while the semantic dependencies offer a simplified view of the native MRS structures[12]. It should be noted that not all linguistic information in the native DeepBank annotations is preserved in the PTB phrase structure and CoNLL dependency formats. Nevertheless, they offer easy access to the data in familiar formats.

We give an example of each of these annotations for a simple sentence from the corpus, beginning with the native derivation which contains sufficient information to enable full reconstruction of the HPSG feature structure returned by the parser. Next is the simplified PTB-style labeled bracketing for the example, then the native semantic representation in MRS, and finally the CoNLL-style dependency view of the syntax and the semantics.

```
(root_strict
 (sb-hd_mc_c
  (hdn_bnp_c
   (aj-hdn_norm_c
    (j-j_crd-att-t_c
     (v_j-nb-pas-tr_dlr (v_pas_odlr (estimate_v4 ("estimated"))))
     (mrk-nh_evnt_c
      (and_conj ("and"))
      (actual_a1 ("actual"))))
    (hdn-aj_rc_c
     (hdn_optcmp_c (n_pl_olr (result_n1 ("results"))))
     (vp_rc-redrel_c
      (hd-cmp_u_c
       (v_prp_olr (involve_v2 ("involving")))
       (hdn_bnp_c (hdn_optcmp_c (n_pl_olr (loss_n1 ("losses"))))))))))
  (hd-cmp_u_c
   (be_c_are ("are"))
   (hd_optcmp_c (w_period_plr (v_pas_odlr (omit_v1 ("omitted."))))))))
```

Figure 1: Sample DeepBank native derivation tree for "*Estimated and actual results involving losses are omitted.*"

In the derivation show in Figure 1, we see that a combination of very general rules and construction-specific ones have been applied to license this analysis: the rule that combines any head with a complement (the *hd-cmp_u_c* rule) is used for the verb phrase "involving losses" and again for "are omitted", while the narrowly constrained rule that converts a VP into a post-nominal modifier (the *vp_rc-redrel_c* rule) is used to ensure the correct semantics for the nominal phrase "results involving losses". The specific lexical entry identifiers are also included in the derivation, showing for example that the entry used for "estimated" here is *estimate_v4*, which happens to be the simple transitive verb, not, say, the raising verb that would be needed for *we estimated there to be dozens of applicants*.

```
(S
 (NP (N (AP (AP (V estimated))
             (AP (CONJ and)
                 (AP actual)))
        (N (N results)
           (S (VP (V involving)
                  (NP (N losses)))))))
 (VP (V are)
     (VP (V omitted.)))))
```

Figure 2: Sample DeepBank PTB-style labeled bracketing for "*Estimated and actual results involving losses are omitted.*"

The simplified view of the syntactic analysis in Figure 2 employs one of a small set of familiar lexical and phrasal category labels for each bracketed constituent. These node labels can be helpful both for cross-framework parser comparisons, and also for coarse-grained searches of the treebank, such as when looking for all noun phrases in a certain configuration, ignoring the internal composition of each NP.

```
<h1,e3:prop:pres:indicative:-:-,
{h4:udef_q<0:45>(x6, h5, h7),
h8:_estimate_v_at<0:9>(e9, i10, x6),
h8:parg_d<0:9>(e11, e9, x6),
h12:_and_c<10:13>(e13, h8, e9, h14, e15),
h14:_actual_a_1<14:20>(e15, x6),
h12:_result_n_of<21:28>(x6:3:pl:+, i16),
h12:_involve_v_1<29:38>(e17, x6, x18),
h19:udef_q<39:45>(x18, h20, h21),
h22:_loss_n_of<39:45>(x18:3:pl:+, i23),
h2:_omit_v_1<50:58>(e3, i24, x6),
h2:parg_d<50:58>(e25, e3, x6)},
{h1 qeq h2, h5 qeq h12, h20 qeq h22}>
```

Figure 3: Sample DeepBank semantics in native MRS representation for "*Estimated and actual results involving losses are omitted.*"

The compact view of the MRS representation shown in Figure 3 employs a strict ascending ordering convention on the arguments for each elementary predication, with the first argument being the inherent variable (a referential index for nominal predications such as *_loss_n_of* and an event variable otherwise). Thus the verbal

ID	FORM	LEMMA	GPOS	HEAD	DEPREL	PRED	ARGS-P1	ARGS-P2	ARGS-P3	ARGS-P4	ARGS-P5	ARGS-P6	ARGS-P7
1	Estimated	estimate	v_np	4	aj-hdn_norm	_estimate_v_at	ARG0	L-INDEX	_	_	_	_	_
2	and	and	c_xp_and	1	j-j_crd-att-t	_and_c	_	ARG0	_	_	_	_	_
3	actual	actual	aj_-_i	2	mrk-nh_evnt	_actual_a_1	_	R-INDEX	ARG0	_	_	_	_
4	results	result	n_pp_c-ns-of	7	sb-hd_mc	_result_n_of	ARG2	_	ARG1	ARG0	ARG1	_	ARG2
5	involving	involve	v_np	4	hdn-aj_rc	_involve_v_1	_	_	_	_	ARG0	_	_
6	losses	loss	n_pp_mc-of	5	hd-cmp_u	_loss_n_of	_	_	_	_	ARG2	ARG0	_
7	are	be	v_prd_are	0	root_strict	_	_	_	_	_	_	_	_
8	omitted.	omit	v_np	7	hd-cmp_u	_omit_v_1	_	_	_	_	_	_	ARG0

Figure 4: Sample DeepBank CoNLL-style dependencies for "*Estimated and actual results involving losses are omitted.*"

predication *_omit_v_1* introduced by the passive "omitted" only has its ARG2 instantiated with the index introduced by "results", leaving its ARG1 uninstantiated, as indicated by the presence of an "i" rather than an "x" variable as the second of its three arguments. Each predication is also marked with a character span from the original sentence, linking this component of the semantics to the corresponding word or phrase that introduced it.

ID	FORM	LEMMA	GPOS	HEAD	DEPREL	PRED	ARGS-P1	ARGS-P2	ARGS-P3	ARGS-P4
1	Estimated	estimate	VBN	4	NMOD	estimate.01	_	AM-ADV	_	_
2	and	and	CC	1	COORD	_	_	_	_	_
3	actual	actual	JJ	2	CONJ	_	_	_	_	_
4	results	result	NNS	7	SBJ	result.01	A1	A2	A2	A1
5	involving	involve	VBG	4	APPO	involve.01	_	_	_	_
6	losses	losses	NNS	5	OBJ	_	_	_	A1	_
7	are	be	VBP	0	ROOT	_	_	_	_	_
8	omitted	omit	VBN	7	VC	omit.01	_	_	_	_
9	.	.	.	7	P	_	_	_	_	_

Figure 5: Sample of original CoNLL (2008) dependencies derived from PTB and PropBank/NomBank annotation

The CoNLL-style dependency format shown in Figure 4 incorporates the essential syntactic and semantic structures of the HPSG analysis in a uniformed token-based dependency representation[5]. The GPOS field contains the "golden" lexical type selected for the corresponding token. The HEAD field records the token ID of the dependency head. The DEPREL is the corresponding dependency type which is inherited from the HPSG rule name. The PRED field contains the name of the elementary predications from the MRS (hence not limited to verbal and nominal predicates). The remaining ARGS fields identify the arguments of each predicate.

In comparison to the PTB + PropBank/NomBank derived dependency annotation for CoNLL Shared Task 2008 (see Figure 5 for an example), the DeepBank data in CoNLL format offers more fine-grained POS and dependency types, and more densely populated semantic graphs. For example, in comparison to the dependency type inventory of [13] used in the CoNLL Shared Tasks which does not distinguish different types of nominal modifiers (NMOD), our dependencies further mark such head-adjunct relations by the type of the modifier being a pre-head adjunct (*aj-hdn_adjn*, as in *"The [big old cat] slept."*), a post-head relative clause (*hdn-aj_rc*, as in *"The [cat we chased] ran."*), or a post-head reduced relative clause (*hdn-aj_redrel*, as in *"A [cat in a tree] fell."*)

[5]Due to the limited page width, not all the columns in the CoNLL 2008 format are shown here.

4 Patching Coverage Gaps with An Approximating PCFG

As we noted above, one potential criticism against a purely grammar-based treebanking approach addresses its lack of complete coverage in analyzing all sentences in the corpus. The missing gaps in coverage are due to one or more of three causes: (i) ill-formed texts as input to the grammar (rare but present); (ii) the lack of linguistic coverage in the grammar implementation (most frequent); or (iii) limits on computing resources – time or memory – imposed in the analysis of any one sentence (perhaps 20% of the failed parses). The first issue is not specific to grammar-based treebanking, and in fact, manual treebanking projects also carefully select (and in many cases edit) the texts to be annotated. The top criterion for the selection step is to keep the meaningful and representative texts while discarding the problematic items for which full linguistic annotation is not worthwhile. For the second and third issues of either incomplete grammar coverage or the lack of efficiency in processing, there is legitimate concern over the robustness of deep linguistic grammars such as the ERG in comparison to creative and flexible human annotators.

In our discriminant-based approach of treebanking, the coverage gap shows up in two ways: either the grammar fails to parse a specific input utterance, or all the candidate analyses proposed by the grammar are rejected through the manual disambiguation step. Both suggest that a desired analysis is missing due to certain constraints in the grammar. Our experience with the Wall Street Journal corpus and the ERG shows that about 8% of the sentences fail to parse, while another 10% received no acceptable analysis despite getting one or more parses from the ERG. In both cases, using the discriminant-based treebanking tools, annotators cannot record an existing good tree for the sentence.

To annotate the sentences in the grammar coverage gap, we use a robust and overgenerating grammar that approximates the parsing behavior of the ERG. More specifically, an approximating probabilistic context-free grammar (PCFG) is extracted from the automatically parsed treebank of the ERG. The categories in the PCFG are the HPSG rule names annotated with additional information either from the syntactic context (derivation tree) or the detailed properties in the feature structure. Due to the unrestrictive nature of the PCFG, it achieves almost full coverage on all the sentences from the original corpus. The approximating PCFG delivers the most likely pseudo-derivations of ERG according to a generative probabilistic model. In combination with the feature structures of the rules and the lexical entries from the orignal ERG, we can recompose the semantics by doing unification on these derivations. In cases where the unification fails, a robust unifier is called instead to override one side of the conflicting constraints according to certain heuristics.

The evaluation shown in [23] suggests that this PCFG, with careful selection of addiontal annotations and the massive automatically created training treebank, achieves very good parsing accuracy. When tested on sentences that the ERG covers correctly, the best PCFG achieved 84.9 (syntactic) ParsEval F1 score, and 84.2 F1 in the semantic argument relation evaluation (EDM_A). Both measures are about

2% lower than the HPSG parser with ERG. The PCFG succeeds in parsing over 99% of the test set, while the original ERG successfully covers about 80% of it. In a comparison to the parsing accuracy of the state-of-the-art Berkeley parser trained with the same corpus, our PCFG training was much more scalable (with up to 50 million automatically ERG parsed trees), yielding much better overall accuracy.

Lastly, we have developed a graphical tree editor that allows the annotators to manually correct the remaining errors in the PCFG parses. The tool not only supports an intuitive drag-and-drop style of editing, but also records the entire editing sequence, creating additional raw annotation data for future research. Preliminary experience on the post-editing steps suggests that an annotator can correct 35-40 sentences per hour, producing for each a derivation tree which contains at least one constituent not (yet) licensed by the ERG, but necessary for the correct analysis of the sentence.

5 Next Steps

Among the principal advantages claimed for this DeepBank approach is the ability to make successive refinements to the treebank annotations, by making changes to the grammar or to the parsing configuration, and then reparsing and updating with the existing discriminant-based annotations. One planned change in that parsing configuration is to record in the database the full (packed) parse forest for each sentence, rather than the 500 highest-ranked parses currently stored. Manual disambiguation from the full forest will require a new treebanking tool, still under development, but initial experiments already confirm that the existing discriminants are sufficient to automatically fully disambiguate the great majority of the previously treebanked WSJ sentences even working with full parse forests. This full-forest method will provide greater stability in the English DeepBank, eliminating the current minor but annoying uncertainty that results from the dependence on parse ranking to preserve the desired analysis among the top-ranked 500.

6 Conclusion

The English DeepBank provides linguistically rich syntactic and semantic annotations grounded in a well-established and leading linguistic theory (HPSG) for a large and familiar corpus, the million-word Wall Street Journal portion also annotated in the Penn Treebank. The first public release of this resource will include manually selected full analyses produced by the English Resource Grammar for more than 80% of these 50,000 sentences, providing unmatched consistency and linguistic detail, available in multiple formats and representations. The remainder of the corpus will be annotated with compatible though approximate syntactic and semantic analyses produced using a PCFG trained on the manually annotated treebank, to ensure complete coverage of the corpus in the treebank. Adopting the Redwoods methodology for constructing and maintaining this dynamic treebank

will enable further improvements in the grammar to be projected into updated versions of the DeepBank, along with correction of any remaining annotation errors. We believe that an annotated resource of this scale for this corpus will be useful for research both in NLP and in corpus-based theoretical work in linguistics and psycholinguistics.

Acknowledgements

The project is partially supported by the Erasmus Mundus European Masters Program in Language and Communication Technologies (http://www.lct-master.org; EM Grant Number: 2007-0060). The second author also thanks the *Deependance* project funded by BMBF (01IW11003) for its support of the work.

References

[1] Peter Adolphs, Stephan Oepen, Ulrich Callmeier, Berthold Crysmann, Dan Flickinger, and Bernd Kiefer. Some fine points of hybrid natural language parsing. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*, Marrakech, Morocco, 2008.

[2] Sabine Brants, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. The tiger treebank. In *Proceedings of the workshop on treebanks and linguistic theories*, pages 24–41, 2002.

[3] Thorsten Brants. TnT - a statistical part-of-speech tagger. In *Proceedings of the 6th Conference on Applied Natural Language Processing (ANLP 2000)*, Seattle, USA, 2000.

[4] Ulrich Callmeier. Efficient parsing with large-scale unification grammars. Master's thesis, Universität des Saarlandes, Saarbrücken, Germany, 2001.

[5] Jean Carletta. Assessing agreement on classification tasks: the kappa statistic. *Computational Linguistics*, 22(2):249–254, 1996.

[6] David Carter. The treebanker: a tool for supervised training of parsed corpora. In *Proceedings of the ACL Workshop on Computational Environments for Grammar Development and Linguistic Engineering*, pages 9–15, Madrid, Spain, 1997.

[7] Ann Copestake. *Implementing Typed Feature Structure Grammars*. CSLI, Stanford, USA, 2002.

[8] Ann Copestake, Dan Flickinger, Carl J. Pollard, and Ivan A. Sag. Minimal recursion semantics: an introduction. *Research on Language and Computation*, 3(4):281–332, 2005.

[9] Stefanie Dipper. Grammar-based corpus annotation. In Anne AbeillÃ©, Thorsten Brants, and Hans Uszkoreit, editors, *Proceedings of the Workshop on Linguistically Interpreted Corpora LINC-2000 , Luxembourg*, pages 56–64, 2000.

[10] Dan Flickinger. On building a more efficient grammar by exploiting types. In Stephan Oepen, Dan Flickinger, Jun'ichi Tsujii, and Hans Uszkoreit, editors, *Collaborative Language Engineering*, pages 1–17. CSLI Publications, 2002.

[11] Jan Hajič, Alena Böhmová, Eva Hajičová, and Barbora Vidová-Hladká. The Prague Dependency Treebank: A Three-Level Annotation Scenario. In A. Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*, pages 103–127. Amsterdam:Kluwer, 2000.

[12] Angelina Ivanova, Stephan Oepen, Lilja Øvrelid, and Dan Flickinger. Who did what to whom? a contrastive study of syntacto-semantic dependencies. In *Proceedings of the Sixth Linguistic Annotation Workshop*, pages 2–11, Jeju, Republic of Korea, 2012.

[13] Richard Johansson and Pierre Nugues. Extended Constituent-to-dependency Conversion for English. In *Proceedings of NODALIDA 2007*, Tartu, Estonia, 2007.

[14] Tracy Holloway King, Richard Crouch, Stefan Riezler, Mary Dalrymple, and Ronald M. Kaplan. The parc 700 dependency bank. In *In Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora (LINC-03*, pages 1–8, 2003.

[15] Robert Malouf, John Carroll, and Ann Copestake. Efficient feature structure operations without compilation. *Natural Language Engineering*, 6:29–46, 2000.

[16] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330, 1993.

[17] Stephan Oepen. [incr tsdb()] — competence and performance laboratory. User manual. Technical report, Computational Linguistics, Saarland University, Saarbrücken, Germany, 2001.

[18] Stephan Oepen, Kristina Toutanova, Stuart Shieber, Christopher Manning, Dan Flickinger, and Thorsten Brants. The LinGO Redwoods treebank: motivation and preliminary applications. In *Proceedings of COLING 2002: The 17th International Conference on Computational Linguistics: Project Notes*, Taipei, Taiwan, 2002.

[19] Carl J. Pollard and Ivan A. Sag. *Head-Driven Phrase Structure Grammar*. University of Chicago Press, Chicago, USA, 1994.

[20] Victoria Rosén, Paul Meurer, and Koenraad de Smedt. LFG Parsebanker: A Toolkit for Building and Searching a Treebank as a Parsed Corpus. In Frank Van Eynde, Anette Frank, Gertjan van Noord, and Koenraad De Smedt, editors, *Proceedings of the 7th International Workshop on Treebanks and Linguistic Theories (TLT7)*, pages 127–133, Utrecht, 2009. LOT.

[21] Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of the 12th Conference on Computational Natural Language Learning (CoNLL-2008)*, Manchester, UK, 2008.

[22] L. van der Beek, G. Bouma, R. Malouf, and G. van Noord. The alpino dependency treebank. In *Computational Linguistics in the Netherlands (CLIN) 2001*, Twente University, 2002.

[23] Yi Zhang and Hans-Ulrich Krieger. Large-scale corpus-driven pcfg approximation of an hpsg. In *Proceedings of the 12th International Conference on Parsing Technologies*, pages 198–208, Dublin, Ireland, 2011.

ParDeepBank: Multiple Parallel Deep Treebanking

Dan Flickinger[1], Valia Kordoni[3,2], Yi Zhang[2],
António Branco[4], Kiril Simov[5], Petya Osenova[5],
Catarina Carvalheiro[4], Francisco Costa[4], Sérgio Castro[4]
[1]Stanford University, USA, [2]DFKI GmbH, Germany,
[3]Humboldt University of Berlin, Germany,
[4]University of Lisbon, Portugal,
[5]Bulgarian Academy of Sciences, Bulgaria
danf@stanford.edu, evangelia.kordoni@anglistik.hu-berlin.de,
yizhang@dfki.de, antonio.branco@di.fc.ul.pt,
kivs@bultreebank.org, petya@bultreebank.org,
catarina.carvalheiro@di.fc.ul.pt, f.costa@di.fc.ul.pt,
sergio.castro@di.fc.ul.pt

Abstract

This paper describes the creation of an innovative and highly parallel treebank of three languages from different language groups — English, Portuguese and Bulgarian. The linguistic analyses for the three languages are done by compatible parallel automatic HPSG grammars using the same formalism, tools and implementation strategy. The final analysis for each sentence in each language consists of (1) a detailed feature structure analysis by the corresponding grammar and (2) derivative information such as derivation trees, constituent trees, dependency trees, and Minimal Recursion Semantics structures. The parallel sentences are extracted from the Penn Treebank and translated into the other languages. The Parallel Deep Bank (ParDeepBank) has potentially many applications: for HPSG grammar development; machine translation; evaluation of parsers on comparable data; etc.

1 Introduction

In this paper we present the initial version of ParDeepBank — a parallel treebank for three languages: English, Portuguese, and Bulgarian. The annotation of each sentence in the treebank is automatically analyzed by a deep HPSG grammar (Head-driven Phrase Structure Grammar: [24]) for the corresponding language. These grammars are implemented within the same linguistic theory and formalism, following a similar approach to grammar implementation. The correct analysis is selected manually in all of the three cases. The HPSG grammars for the three languages have different levels of development. Hence, they differ in their coverage. In order to process the whole set of sentences in the parallel treebank,

we employ all the available NLP tools for each language and produce a comparable analysis for each sentence. The sentences are selected from PTB (PennTreebank) data and then translated to Portuguese and Bulgarian. Our parallel treebanking approach takes a starting point and motivation similar to that already adopted for the Czech-English PennTreebank Corpus[1].

Recently, a number of initiatives have been observed for constructing parallel treebanks. For example, in [19] the construction of a parallel Czech-Russian Dependency Treebank required smoothing of the syntactic schemes before handling the alignments on various levels. Another project, among others, is SMULTRON: a parallel treebank of English, German and Swedish [32]. In this case the annotation scheme is different for each treebank. The alignments are done on the sentence, phrase and word level. There are attempts for construction of parallel treebanks on a large scale and in a fully automated way [31], where large multilingual corpora are parsed with the respective language tools, and then models are created for alignments using small amounts of aligned data as well as complex features.

We consider our efforts to be innovative in a number of directions, including performing multilingual treebanking in the same semantic formalism (MRS: [13]), which ensures a deep logical representation for the syntactic analyses and ensures automatic alignments, which would make possible the comparison of semantic structures among parallel data in different languages. Our approach would facilitate the incorporation of further languages to the ParDeepBank, since it allows for a multilayered addition of the parser as well as the analysed data. The English DeepBank and ERG play the role of state-of-the-art pivots with respect to the core language phenomena coverage and best parsing abilities. The other grammars and treebanks aim at the established standards, but their development is supported by additional robustness mechanisms, such as dependency analyses plus rule-based projection to MRS in the case of the Bulgarian Grammar. The sentences, analyzed via a dependency parser, will be processed additionally by the BURGER grammar when it is extended appropriately. The current manually checked dependency analyses will be used for the selection of the correct analyses, produced by BURGER. Similarly, such analyses updates might be expected for the other languages, too. Our design is as follows: we use parallel data from the Wall Street Journal portion of the Penn Treebank, then parse it, using common NLP components, with each of the three grammars, which are represented in an identical formalism on syntactic and semantic grounds. Our initial goal is each sentence of WSJ corpus to be analyzed syntactically and semantically in MRS.

The structure of the paper is as follows: Section 2 introduces the idea behind the ParDeepBank; Section 3 describes the Resource Grammars for three languages: English, Portuguese and Bulgarian; Section 4 focuses on the process of sentence selection and the details of treebanking in English, Portuguese and Bulgarian; Section 5 outlines some preliminary features of the dynamic parallel treebanking; Section 6 concludes the paper and presents insights on future work.

[1] http://ufal.mff.cuni.cz/pcedt1.0/doc/PCEDT_body.html

2 The ParDeepBank

The PTB has emerged as the de facto standard for evaluation of different NLP analyzers for English including POS taggers, chunkers, and parsers (both constituent and dependency). Even for non-English-oriented analyzers there is a need for comparable methods of evaluation. The PTB is also widely used for creation of new resources like the Penn Discourse Treebank [25]. It is a natural step to reuse the same text in the process of creation of deep processed corpora. The treebanking for the ParDeepBank is built on the same approach used for the soon-to-be-released English DeepBank, which adopts the Redwoods treebanking approach of [22]. The annotation of a sentence starts with producing all possible analyses with respect to the English Resource Grammar (ERG [16]). The system calculates the set of binary discriminants which disambiguate between the different analyses of each sentence. These discriminants are used by the annotators to select the correct analysis. There are two cases when a given sentence is not included in the DeepBank: (1) when the ERG fails to produce any analysis of the sentence at all, and (2) when the annotator cannot find a correct analysis among the candidates. In both cases a modification of the ERG would be necessary in order to produce the required analysis. In the current version of the English DeepBank development, some 92% of all sentences in the WSJ section of the PTB receive one or more candidate analyses, and 82% of all sentences are successfully annotated with the correct analysis in the DeepBank.

The creation of ParDeepBank extends the work on the English DeepBank in the multilingual dimension. This effort is necessary for several reasons:

- *Comparable evaluation of NLP tools for several languages.* In many cases, NLP systems exploit hybrid architectures which include language-specific components that are hard to transfer to other languages. Thus, application of the same system as the one used for English is expensive, if not impossible. In such cases, if a given work reports 97.83 % accuracy for a POS tagger of Bulgarian, it is not possible to compare it to an English POS tagger reporting 98.03 % accuracy, for the evaluation corpora are not comparable in any sense. The construction of ParDeepBank will overcome this problem by providing directly comparable analysis on several linguistic levels for several languages over parallel texts. The treebank might be used for defining comparable measures for various languages and different NLP tasks.

- *Comparable coverage of the resource grammars for several languages.* Although the development of most of the resource grammars for different languages follows similar scenarios, their coverage diverges in the process of development. ParDeepBank will provide a basis for measuring the coverage of such grammars over a large amount of parallel data. This is of great importance with respect to exploitation of such grammars in applications like machine translation (see [2]).

- *Linguistic research.* Parallel corpora annotated with such detailed linguistic

analyses are valuable for language data in multilingual contexts. The selection of the three languages in different language families will also facilitate the comparison of language phenomena.

At the moment the idea behind ParDeepBank is demonstrated through an extension of DeepBank with two other languages: Portuguese and Bulgarian. For both languages a portion of the data present in the DeepBank has been translated, parsed with the two other grammars and parallelized. The details for each language effort are presented and commented on in the next sections.

3 Resource Grammars for Three Languages

3.1 English Resource Grammar

The ERG is an open-source, broad-coverage, declarative grammar implementation for English, designed within the HPSG framework. This linguistic framework places most of the burden of linguistic description on the lexicon, employing a relatively small number of highly schematic syntactic rules to combine words or phrases to form larger constituents. Each word or phrase (more generally, each sign) is defined in terms of feature structures where the values of attributes are governed by general principles such as the Head Feature Convention, which ensures the identity of a particular subset of feature-value pairs on a phrasal node and on one distinguished daughter, the head of the phrase. Many of these generalizations aim to be language-independent, constraining the space of possible analyses for linguistic phenomena. Central to the HPSG framework is the inclusion in each sign of constraints on multiple dimensions of representation, including at least syntactic and semantic properties, so that rules, lexical entries, and principles determine semantic well-formedness just as they do syntax. Under continuous development at CSLI since 1993, the ERG provides syntactic and semantic analyses for the large majority of common constructions in written English text (cf. [17]). The current grammar consists of a 35,000-word lexicon instantiating 980 leaf lexical types, as well as 70 derivational and inflection rules, and 220 syntactic rules.

3.2 Portuguese Resource Grammar

The development of the Portuguese part of the ParDeepBank was supported by the Portuguese Resource Grammar LXGram. This grammar was presented at length in [7], [8]. A brief overview is provided in the present section. LXGram is based on hand coded linguistic generalizations supplemented with a stochastic model for ambiguity resolution. Like the other grammars used for the English and Bulgarian parts of the ParDeepBank, it follows the grammatical framework of Head-Driven Phrase Structure Grammar (HPSG, [24]). As this is a linguistic framework for which there is a substantial amount of published work, this option allows for the straightforward implementation of grammatically principled analyses that have

undergone extensive scientific scrutiny. Following this grammatical framework, LXGram associates grammatical representations to natural language expressions, including the formal representation of their meaning, thus providing for so called deep linguistic processing of the input sentences. It was developed on top of a cross-language seed computational grammar fostered by the Matrix project [1].

Like several other computational grammars, including the other two used for the construction of the present multilingual ParDeepBank, LXGram uses Minimal Recursion Semantics (MRS, [13]) for the representation of meaning. An MRS representation supports scope underspecification; i.e. it is a description of a set of possible logic formulas that differ only in the relative scope of the relations present in these formulas. This format of semantic representation is well defined in the sense that it is known how to map between MRS representations and formulas of second order logic, for which there is a set-theoretic interpretation.

LXGram is developed in the Linguistic Knowledge Builder (LKB) system [11], an open-source development environment for constraint-based grammars. This environment provides a GUI, debugging tools and very efficient algorithms for parsing and generation with the grammars developed there. Several broad coverage HPSGs have been developed in the LKB, of which the largest ones are for English [12], used in this paper, German [14] and Japanese [26].

LXGram is in active development, and it already encompasses a wide range of linguistic phenomena, such as long distance dependencies, coordination, subordination, modification and many subcategorization frames, with a lexicon containing around 25 000 entries. In its last stable version, it contains over 60 lexical rules, 100 syntax rules, and 850 lexical leaf types (determining syntactic and semantic properties of lexical entries). It resorts to a pre-processing step performed by a pipeline of the shallow processing tools handling tokenization, POS tagging, morphological analysis, lemmatization and named entity recognition [3], [4], [15], [10].

LXGram copes with the European and the Brazilian variants of Portuguese. It contains lexical entries that are specific to either of them, and it covers both European and Brazilian syntax, as more thoroughly described in [5], [6]. The LXGram operation is coupled with a statistical disambiguation model, in order to automatically select the most likely analysis of a sentence when the grammar produces multiple solutions. Using a maximum entropy algorithm, this model is trained from the CINTIL DeepBank [9]. At present, this dataset comprises over 10 000 sentences of newspaper text, and development continues. The linguistic analyses that are implemented in the grammar are documented in a report that is updated and expanded with each version of the grammar. The grammar is available for download at http://nlx.di.fc.ul.pt/lxgram, together with this documentation.

An experiment was conducted to assess the coverage of LXGram's version on spontaneous text at the time of the experiment. This experiment and its results are presented at length in [8]. In a nutshell, the grammar exhibited a parsing coverage of around one third (i.e. one third of the input sentences get at least one parse by the grammar), and a parsing accuracy in the range of 30-40% (i.e. from the sentences that got at least one parse, that was the proportion of sentences for which

the grammar delivers the correct grammatical representation).[2]

3.3 Bulgarian Resource Grammar

In the development of the Bulgarian part of ParDeepBank we rely on the Bulgarian HPSG resource grammar BURGER [23], and on a dependency parser (Malt Parser — [20], trained on the BulTreeBank data. Both parsers produce semantic representations in terms of MRS. BURGER automatically constructs them, while the output of the Malt Parser is augmented with rules for construction of MRS-es from the dependency trees. The integration of both tools has several advantages, such as: in the first version of the Bulgarian part of the parallel treebank, all the translated from English sentences have a correct analysis on MRS level, which to be used for the alignment purposes. Later on, when BURGER is extended to cover also these sentences, the analyses will be substituted. BURGER covers the main constructions in the MRS dataset of ERG (translated into Bulgarian and augmented with some sentences from BulTreeBank). Then it has been extended by a verbal lexicon, containing about 15000 verb lemmas (more than 700000 wordforms) encoded on morphosyntactic level as well as about 3000 ones, encoded on valency level. We are working on the extension of the lexicon of BURGER with more valency information and other parts-of-speech entries.

The chosen procedure, as mentioned above, is as follows: first, the Bulgarian sentences are parsed with BURGER. If it succeeds, then the produced MRS-es are used for the alignment. In case BURGER has no coverage, the sentences are parsed with the Malt Parser, and then MRS-es are constructed over the dependency parses. The MRS-es are created via a set of transformation rules [27]. Here is an overview of the components of the Bulgarian Language Processing Pipeline, exploited within the work:

- POS tagging. POS tagging is performed by a cascade of several modules — including a guesser, linguistic knowledge (lexicon and rules) and a statistical tagger. The accuracy of the whole pipeline is 97.83% — [18]. In this pipeline the SVM POS Tagger plays the role of a guesser for the GTagger.

- Lemmatization. The lemmatization is based on the morphological lexicon. From the lexicon we extracted functions which convert each wordform into its basic form (as a representative of the lemma). The accuracy is 95.23%.

- Dependency parsing. MALTParser has been trained on the dependency version of BulTreeBank. The model achieves 87.6% labeled parsing accuracy.

- MRS analyzer. We exploit two types of rules over the dependency parses. The first constructs for each lexical node its elementary predication, while

[2]To put these results into perspective, it is worth mentioning [33], who report values of 80.4% parsing coverage on newspaper text for ERG, 42.7% for the Japanese grammar and 28.6% for the German grammar, which have been in development for over 15 years now, being older than LXGram, with around 5 years of development.

the second combines the structures for two nodes on the basis of the dependency relations.

4 Sentence Selection and Treebanking

English DeepBank The English DeepBank is a treebank created by application of the Redwoods treebank approach to the Wall Street Journal (WSJ) corpus included in the PTB. The process of DeepBank annotation of the WSJ corpus is organised into iterations of a cycle of parsing, treebanking, error analysis and grammar/treebank updates, with the goal of maximizing the accuracy of annotation through successive refinement.

Sentences from the WSJ are first parsed with the PET parser using the ERG. Up to 500 top readings are recorded for each sentence. The exact best-first parsing mode guarantees that these recorded readings are the ones that have "achieved" the highest disambiguation scores according to the currently in-use parse selection model, without enumerating through all possible analyses.

The parsing results are then manually disambiguated by the human annotators. However, instead of looking at individual trees, the annotators spend most of their effort making binary decisions on either accepting or rejecting constructions. Each of these decisions, called discriminants, reduces the number of the trees satisfying the constraints (here maybe an example is due). Every time a decision is made, the remaining set of trees and discriminants are updated simultaneously. This continues until one of the following conditions is met: i) if there is only one remaining tree and it represents a correct analysis of the sentence, the tree is marked as gold; ii) if none of the remaining trees represents a valid analysis, the sentence will be marked as "rejected", indicating an error in the grammar[3]; iii) if the annotator is not sure about any further decision, a "low confidence" state will be marked on the sentence, saved together with the partial disambiguation decisions. Generally speaking, given n candidate trees, on average $\log_2 n$ decisions are needed in order to fully disambiguate. Given that we set a limit of 500 candidate readings per sentence, the whole process should require no more than 9 decisions. If both the syntactic and the semantic analyses look valid, the tree is recorded as the gold reading for the sentence.

While the grammar development is independent to the treebanking progress, we periodically incorporate the recent changes of the grammar into the treebank annotation cycle. When a grammar update is incorporated, the treebank also gets updated accordingly by i) parsing anew all the sentences with the new grammar; ii) re-applying the recorded annotation decisions; iii) re-annotating those sentences which are not fully disambiguated after step ii. The extra manual annotation effort in treebank update is usually small when compared to the first round annotation.

[3]In some cases, the grammar does produce a valid reading, but the disambiguation model fails to rank it among the top 500 candidates. In practice, we find such errors occuring frequently during the first annotation circle, but they diminish quickly when the disambiguation model gets updated.

Portuguese Component A first step in the construction of the Portuguese part of the ParDeepBank consisted in obtaining a corpus of raw sentences that is parallel to the WSJ corpus, on which the PTB is based. To this end, the WSJ text was translated from English into Portuguese. This translation was performed by two professional translators. Each portion of the corpus that was translated by one of the translators was subsequently reviewed by the other translator in order to double-check their outcome and enforce consistency among the translators.

Given that the original English corpus results from the gathering of newspaper texts, more specifically from the Wall Street Journal, a newspaper specialized on economic and business matters, the translators were instructed to perform the translation as if the result of their work was to be published in a Portuguese newspaper of a similar type, suitable to be read by native speakers of Portuguese. A second recommendation was that each English sentence should be translated into a Portuguese sentence if possible and if that would not clash with the first recommendation concerning the "naturalness" of the outcome.

As the Portuguese corpus was obtained, it entered a process of dynamic annotation, analogous to the one applied to the Redwoods treebank. With the support of the annotation environment [incr tsdb()] [21], the Portuguese Resource Grammar LXGram was used to support the association of sentences with their deep grammatical representation. For each sentence the grammar provides a parse forest; the correct parse if available, can be selected and stored by deciding on a number of binary semantic discriminants that differentiate the different parses in the respective parse forest.

The translation of the WSJ corpus into Portuguese is completed, and at the time of writing the present paper, only a portion of these sentences had been parsed and annotated. While this is an ongoing endeavor, at present the Portuguese part of the ParDeepBank includes more than 1,000 sentences.

The sentences are treebanked resorting to the annotation methodology that has been deemed in the literature as better ensuring the reliability of the dataset produced. They are submitted to a process of double blind annotation followed by adjudication. Two different annotators annotate each sentence in an independent way, without having access to each other's decisions. For those sentences over whose annotation they happen to disagree, a third element of the annotation team, the adjudicator, decides which one of the two different grammatical representations for the same sentence, if any, is the suitable one to be stored. The annotation team consists of experts graduated in Linguistics or Language studies, specifically hired to perform the annotation task on a full time basis.

Bulgarian Component Bulgarian part of ParDeepBank was produced in a similar way to the Portuguese part. First, translations of WSJ texts were performed in two steps. During the first step the text was translated by one translator. We could not afford professional translators. We have hired three students in translation studies(one PhD student and two master students studying at English Department of the

Sofia University). Each sentence was translated by one of them. The distribution of sentences included whole articles. The reason for this is the fact that one of the main problems during the translation turned out to be the named entities in the text. Bulgarian news tradition changed a lot in the last decades moving from transliteration of foreign names to acceptance of some names in their original form. Because of the absence of strict rules, we have asked the translators to do search over the web for existing transliteration of a given name in the original text. If such did not exist, the translator had two possibilities: (1) to transliterate the name according to its pronunciation in English; or (2) to keep the original form in English. The first option was mainly used for people and location names. The second was more appropriate for acronyms. Translating a whole article ensured that the names have been handled in the same way. Additionally, the translators had to translate each original sentence into just one sentence in Bulgarian, in spite of its complex structure. The second phase of the translation process is the editing of the translations by a professional editor. The idea is the editor to ensure the "naturalness" of the text. The editor also has a very good knowledge of English. At the moment we have translated section 00 to 03.

The treebanking is done in two complementary ways. First, the sentences are processed by BURGER. If the sentence is parsed, then the resulting parses are loaded in the environment [incrs tsdb()] and the selection of the correct analysis is done similarly to English and Portuguese cases. If the sentence cannot be processed by BURGER or all analyses produced by BURGER are not acceptable, then the sentence is processed by the Bulgarian language pipeline. It always produces some analysis, but in some cases it contains errors. All the results from the pipeline are manually checked via the visualization tool within the CLaRK system. After the corrections have been repaired, the MRS analyzer is applied. For the creation of the current version of ParDeepBank we have concentrated on the intersection between the English DeepBank and Portuguese DeepBank. At the moment we have processed 328 sentences from this intersection.

5 Dynamic Parallel Treebanking

Having dynamic treebanks as components of ParDeepBank we cannot rely on fixed parallelism on the level of syntactic and semantic structures. They are subject to change during the further development of the resource grammars for the three languages. Thus, similarly to [28] we rely on alignment done on the sentence and word levels. The sentence level alignment is ensured by the mechanisms of translation of the original data from English to Portuguese and Bulgarian. The word level could be done in different ways as described in this section.

For Bulgarian we are following the word level alignment guidelines presented in [29] and [30]. The rules follow the guidelines for segmentation of BulTreeBank. This ensures a good interaction between the word level alignment and the parsing mechanism for Bulgarian. The rules are tested by manual annotation of several

parallel Bulgarian/English corpora which represent the gold standard corpus for word level alignment between Bulgarian and English. For the rest of the data we are following the approach, undertaken for the alignment of the Portuguese DeepBank to English DeepBank.

The word level alignment between Portuguese and English is done in two steps. First, GIZA++ tool[4] is trained on the parallel corpus resulting from the translation of WSJ into Portuguese. This training produces an alignment model which is tuned with respect to this particular project. After this automatic step, a manual checking with a correction procedure is performed. The manual step was performed by the alignment editing tool in the Sanchay collection of NLP tools[5].

The alignment on the syntactic and semantic levels is dynamically constructed from the sentence and word level alignment as well as the current analyses of the sentences in ParDeepBank. The syntactic and semantic analyses in all three treebanks are lexicalized, thus the word level alignment is a good starting point for establishing of alignment also on the upper two levels. As one can see in the guidelines for Bulgarian/English word level alignment, the non-compositional phrases (i.e. idioms and collocations) are aligned on word level. Thus, their special syntax and semantics is captured already on this first level of alignment. Then, considering larger phrases, we establish syntactic and semantic alignment between the corresponding analyses only if their lexical realizations in the sentence are aligned on word level.

This mechanism for alignment on different levels has at least two advantages: (1) it allows the exploitation of word level alignment which is easier for the annotators; and (2) it provides a flexible way for updating of the existing syntactic and semantic alignments when the DeepBank for one or more languages is updated after an improvement has been made in the corresponding grammar. In this way, we have adopted the idea of dynamic treebanking to the parallel treebanking. It provides an easy way for improving the quality of the linguistic knowledge encoded in the parallel treebanks. Also, this mechanism of alignment facilitates the additions of DeepBanks for other languages or additions of analyses in other formalisms.

6 Conclusion and Future Work

In this paper we presented the design and initial implementation of a scalable approach to parallel deep treebanking in a constraint-based formalism, beginning with three languages for which well-developed grammars already exist. The methods for translation and alignment at several levels of linguistic representation are viable, and our experience confirms that the monolingual deep treebanking methodology can be extended quite successfully in a parallel multi-lingual context, when using the same data and the same grammar architecture. The next steps of the ParDeepBank development are the expansion of the volume of aligned annotated

[4]http://code.google.com/p/giza-pp/

[5]http://sanchay.co.in/

data, and improvements in the infrastructure for supporting the creation, maintenance and exploitation of these dynamic Parallel DeepBanks.

References

[1] Bender, E. M., Flickinger, D., and Oepen, S. 2002. The Grammar Matrix. An Open-Source Starter-Kit for the Rapid Development of Cross-Linguistically Consistent Broad-Coverage Precision Grammar. In *Proc. of the Workshop on Grammar Engineering and Evaluation at COLING'02*.

[2] Bond, F., Oepen, S., Siegel, M., Copestake, A., and Flickinger, D. 2005. Open-Source Machine Translation with DELPH-IN. In *Proc. of the Open-Source Machine Translation Workshop at the 10th MTS*, pp. 15–22.

[3] Branco, A., and Silva, J. 2006a. Dedicated Nominal Featurization of Portuguese. In *Proc. of the 7th International Conference on Computational Processing of the Portuguese Language*, PROPOR'06, pp. 244–247.

[4] Branco, A., and Silva, J. 2006b. A Suite of Shallow Processing Tools for Portuguese: LX-Suite. In *Proc. of the Eleventh Conference of the EACL: Posters & Demonstrations*, EACL'06, pp. 179–182.

[5] Branco, A., and Costa, F. 2007a. Accommodating Language Variation in Deep Processing. In *Proc. of GEAF07 Workshop*, pp. 67–86.

[6] Branco, A., and Costa, F. 2007b Self- or Pre-Tuning? Deep Linguistic Processing of Language Variants. In *Proc. of the ACL Workshop on Deep Linguistic Processing*.

[7] Branco, A., and Costa, F. 2008. LXGram in the Shared Task "Comparing Semantic Representations" of STEP2008. In *Proc. of the 2008 Conference on Semantics in Text Processing*, pp. 299–310.

[8] Branco, A., and Costa, F. 2010. LXGram: A Deep Linguistic Processing Grammar for Portuguese. In *LNAI, 6001*, pp. 86–89.

[9] Branco, A., Costa, F., Silva, J., Silveira, S., Castro, S., Avelãs, M., Pinto, C., and Graça, J. 2010. Developing a Deep Linguistic Databank Supporting a Collection of Treebanks: CINTIL DeepGramBank. In *Proc. of LREC'10*.

[10] Branco, A., and Nunes, F. 2012. Verb Analysis in a Highly Inflective Language with an MFF Algorithm. In *Proc. of the 10th international conference on Computational Processing of the Portuguese Language*, PROPOR'12, pp. 1–11.

[11] Copestake, A. 2002. *Implementing Typed Feature Structure Grammars*. CSLI Publications.

[12] Copestake, A., and Flickinger, D. 2000. An Open-Source Grammar Development Environment and Broad-Coverage English Grammar Using HPSG. In *Proc. of the LREC00.*

[13] Copestake, A., Flickinger, D., Pollard, C., and Sag, I. 2005. Minimal Recursion Semantics: an Introduction. *Research on Language and Computation*, 3(4).

[14] Crysmann, B. 2007. Local Ambiguity Packing and Discontinuity in German. In *Proc. of the ACL Workshop on Deep Linguistic Processing*.

[15] Ferreira, E., Balsa, J., and Branco, A. 2007. Combining Rule-Based and Statistical Methods for Named Entity Recognition in Portuguese. In *Proc. of the 5th Workshop em Tecnologia da Informação e da Linguagem Humana*, pp. 1615–1624.

[16] Flickinger, D. 2002. On Building a More Efficient Grammar by Exploiting Types. In *Collaborative Language Engineering*, pp. 1–17. CSLI Publications.

[17] Flickinger, D. 2011. Accuracy vs. Robustness in Grammar Engineering. In *Language from a Cognitive Perspective: Grammar, Usage, and Processing*, pp. 31–50. CSLI Publications.

[18] Georgiev, G., Zhikov, V., Osenova, P., Simov, K., and Nakov, P. 2012. Feature-Rich Part-Of-Speech Tagging for Morphologically Complex Languages: Application to Bulgarian. In *EACL 2012*.

[19] Klyueva, N., and Mareček, D. 2010. Towards parallel czech-russian dependency treebank. In *Proc. of the Workshop on Annotation and Exploitation of Parallel Corpora*.

[20] Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, E., Kübler, S., Marinov, S., and Marsi, E. 2007. MaltParser: A Language-Independent System for Data-Driven Dependency Parsing. In *Natural Language Engineering, 13(2)*, pp. 95–135.

[21] Oepen, S. 1999. *[incr tsdb()] - Competence and Performance Laboratory*. Saarland University.

[22] Oepen, S., Toutanova, K., Shieber, S., Manning, C., Flickinger, D., and Brants, T. 2002. The LinGO Redwoods Treebank: Motivation and Preliminary Applications. In *Proc. of COLING'02*, pp. 1–5.

[23] Osenova, P. 2010. *The Bulgarian Resource Grammar*. VDM.

[24] Pollard, C., and Sag, I. 1994. *Head-Driven Phrase Structure Grammar*. Studies in Contemporary Linguistics. University of Chicago Press.

[25] Prasad, R., Dinesh, N., Lee, A., Miltsakaki E., Robaldo, L., Joshi, A., and Webber, B. 2008. The Penn Discourse Treebank 2.0. In *In Proc. of LREC'08*.

[26] Siegel, M., and Bender, E. 2002. Efficient Deep Processing of Japanese. In *Proc. of COLING'02*.

[27] Simov, K., and Osenova, P. 2011. Towards Minimal Recursion Semantics over Bulgarian Dependency Parsing. In *Proc. of the RANLP 2011*.

[28] Simov, K., and Osenova, O. 2012. Bulgarian-English Treebank: Design and Implementation. In *Proc. TLT10*.

[29] Simov, K., Osenova, P., Laskova, L., Kancheva, S., Savkov, A., and Wang, R. 2012. HPSG-Based Bulgarian-English Statistical Machine Translation. *Littera et Lingua*.

[30] Simov, K., Osenova, P., Laskova, L., Savkov, A., and Kancheva, S. 2011. Bulgarian-English Parallel Treebank: Word and Semantic Level Alignment. In *Proc. of The Second Workshop on Annotation and Exploitation of Parallel Corpora*, pp. 29–38.

[31] Tiedemann, J., and Kotzé, G. 2009. Building a Large Machine-Aligned Parallel Treebank. In *Proc. of TLT08*, pp. 197–208.

[32] Volk, M., Göhring, A., Marek, T., and Samuelsson, Y. 2010. SMULTRON (version 3.0) — The Stockholm MULtilingual Parallel TReebank.

[33] Zhang, Y., Wang, R., and Oepen S. 2009. Hybrid Multilingual Parsing with HPSG for SRL. In *Proc. of CoNLL 2009*.

The Effect of Treebank Annotation Granularity on Parsing: A Comparative Study

Masood Ghayoomi
Freie Universität Berlin
E-mail: masood.ghayoomi@fu-berlin.de

Omid Moradiannasab
Iran University of Science and Technology
E-mail: omidmoradiannasab@gmail.com

Abstract

Statistical parsers need annotated data for training. Depending on the available linguistic information in the training data, the performance of the parsers vary. In this paper, we study the effect of annotation granularity on parsing from three points of views: lexicon, part-of-speech tag, and phrase structure. The results show that changing annotation granularity at each of these dimensions has a significant impact on parsing performance.

1 Introduction

Parsing is one of the main tasks in Natural Language Processing (NLP). The state-of-the-art statistical parsers are trained with treebanks [4, 5], mainly developed based on the Phrase Structure Grammar (PSG). The part-of-speech (POS) tags of the words in the treebanks are defined according to a tag set which contains the syntactic categories of the words with the optional additional morpho-syntactic information. Moreover, the constituent labels in treebanks might also be enriched with syntactic functions. The developed annotated data in the framework of deeper formalisms such as HPSG [13] has provided a fine representation of linguistic knowledge. The performance of the parsers trained with the latter data set have not beaten the state-of-the-art results [12] which shows that fine-grained representation of linguistic knowledge adds complexities to a parser and it has a counter-effect on the performance of the parser. In this paper, we aim to comprehensively study the effect of annotation granularity on parsing from three points of views: lexicon, POS tag, and phrase structure. This study has a different perspective than Rehbein and van Genabith [14]. We selected Persian, a language from the Indo-European language family, as the target language of our study.

2 Treebank Annotation Dimensions

Lexicon: The words of a language represent fine-grained concepts, and the linguistic information added to the words plays a very important role for lexicalized,

statistical parsers. Since data sparsity is the biggest challenge in data oriented parsing, parsers will have a poor performance if they are trained with a small set of data, or when the domain of the training and the test data are not similar. Brown et al. [2] pioneered to use word clustering for language modeling methods. Later on, word clustering was widely used in various NLP applications including parsing [3, 7, 9]. The Brown word clustering is an approach which provides a coarse level of word representation such that the words which have similarities with each other are assigned to one cluster. In this type of annotation, instead of words, the corresponding mapped clusters are used.

POS Tag: The syntactic functions of words at the sentence level are the very basic linguistic knowledge that the parser learns; therefore, they play a very important role on the performance of a parser. The quality of the assigned POS tags to the words and the amount of information they contain have a direct effect on the performance of the parser. The representation of this knowledge can be either coarse-grained such as Noun, Verb, Adjective, etc, or fine-grained to contain morpho-syntactic and semantic information such as Noun-Single, Verb-Past, Adjective-superlative, etc. The fine representation of the tags leads to increase the tag set size and intensify the complexity of the tagging task for a statistical POS tagger to disambiguate the correct labels.

Phrase Structure: Depending on the formalism used as the backbone of a treebank, the labels of the nodes at the phrasal level can be either fine- or coarse-grained. The annotated data in the Penn Treebank [11] provides relatively coarse-grained phrase structures in which mostly the types of the phrasal constituents such as NP, VP, etc are determined. It is not denied that in the latest version of the Penn Treebank the syntactic functions are added to the labels as well, but this information is not available for all nodes. Contrary, annotated data of deep formalisms like HPSG provides a fine representation of phrase structures since the types of head-daughters' dependencies are defined explicitly for all nodes such as the Bulgarian treebank [15] and the Persian treebank [6]. Representation of dependency information on constituents adds complexities to the parser for disambiguating the type of the dependencies as the size of the constituent label set is increased.

3 Evaluation

3.1 Data Set and Tool

The Bijankhan Corpus[1] contains more than 2.5 million word tokens, and it is POS tagged manually with a rich set of 586 tags containing morpho-syntactic and semantic information [1] such that there is a hierarchy on the assigned tags based on the EAGLES guidelines [10]. The number of the main syntactic categories in the tag set is 14, and it is increased to 15 when clitics are splited from their hosts.

The Persian Treebank (PerTreeBank)[2] is a treebank for Persian developed in the framework of HPSG [13] such that the basic properties of HPSG are simulated

[1] http://ece.ut.ac.ir/dbrg/bijankhan/

[2] http://hpsg.fu-berlin.de/∼ghayoomi/PTB.html

but without feature structures. This treebank contains 1016 trees with 27659 word tokens from the Bijankhan Corpus, and it is developed semi-automatically via a bootstrapping approach [6]. In this treebank, the types of head-daughter dependencies are defined according to the HPSG basic schemas, namely head-subject, head-complement, head-adjunct, and head-filler; therefore it is a hierarchical treebank which represents subcategorization requirements. Moreover, the trace for scrambled or extraposed elements and also empty nodes for ellipses are explicitly determined.

Stanford Parser is the Java implementation of a lexicalized, probabilistic natural language parser [8]. The parser is based on an optimized Probabilistic Context Free Grammar (PCFG) and lexicalized dependency parsers, and a lexicalized PCFG parser. Based on the study of Collins [5], heads should be provided for the parser. This has been done semi-automatically for Persian based on the head-daughters' labels. The evaluation of the parsing results is done with Evalb[3] to report the standard bracketing metric results like precision, recall, and F-measure.

SRILM Toolkit contains the implementation of the the Brown word clustering algorithm [2] and it is used to cluster the lexical items in the Bijankhan Corpus [16].

3.2 Setup of the Experiments

Section 2 described the three possible annotation dimensions for parsing. In the followings, we will describe the data preparation and the setup of our experiments to study the effect of each dimension's annotation on parsing performance.

Besides of preparing the PerTreeBank based on the Penn Treebank style explained in Ghayoomi [7], we modified the treebank in three directions for our experiments. The SRILM toolkit is used to cluster the words in the Bijankhan Corpus. Since the Brown algorithm requires a predefined number of clusters, we set the number of clusters to 700 based on the extended model of Brown algorithm proposed by Ghayoomi [7] to treat homographs distinctly. In order to provide a coarse-grained representation of morpho-syntactic information of the words, only the main POS tags of the words (the 15 tags) are used instead of all 586 tags; and in order to provide simple head-daughter relations as coarse-grained phrase structures, only the type of dependencies on adjunct-daughters and complement-daughters as well as the type of clauses on marker-daughters are removed without any changes on other head-daughter relations.

In each model, we train the Stanford parser with the Persian data. Since PerTreeBank is currently the only available annotated data set for Persian with constituents, this data set is used for both training and testing. The 10-fold cross validation is performed to avoid any overlap between the two data sets.

3.3 Results

In the first step of our experiments, we trained the Stanford parser with PerTreeBank without any changes on annotation granularities (Model 1) and recognized it as the baseline model. To further study the effect of each annotation dimension,

[3] http://nlp.cs.nyu.edu/evalb/

Table 1: The parsing results for applying different annotation dimensions

Model	Annotation Dimension (Lexicon,POS Tag,Phrase Structure)			F-score	Precision	Recall
Model 1	Word	Fine	Fine	50.09	50.20	49.99
Model 2	Class	Fine	Fine	55.90	55.95	55.86
Model 3	Word	Coarse	Fine	41.32	41.14	41.51
Model 4	Word	Fine	Coarse	55.13	55.24	55.03
Model 5	Word	Coarse	Coarse	45.34	45.30	45.38
Model 6	Class	Fine	Coarse	59.73	59.81	59.64
Model 7	Class	Coarse	Fine	54.31	54.25	54.38
Model 8	Class	Coarse	Coarse	57.37	57.41	57.34

we did our experiments in three steps such that in each step only one dimension is focused. The fine- vs coarse-grained variabilities at each of the annotation dimensions resulted in eight possible configurations of which the obtained results are reported in Table 1.

To determine the effect of data sparsity at the lexical level, we examined a class-based model with fine-grained POS tag and phrase structure annotations (Model 2). Comparing the results with the baseline, the class-based parsing outperforms the word-based model which indicates the negative impact of data sparsity and the superiority of coarse-grained lexicon representation on parsing.

To find out the effect of detailed morpho-syntactic information on parsing, we built a model such that the POS tags are coarse-grained but the lexicon and phrase structure are fine-grained (Model 3). Comparing this model with the baseline, there is a significant drop on the performance of the parser which indicates that the detailed morpho-syntactic information in the POS tags plays a very important role on parsing. Even though fine-grained POS tags increase the complexity of the tagging task, they have a positive impact on parsing performance because of using detailed information for defining the dependencies.

To study the effect of the HPSG-based annotation on parsing, we built a model (Model 4) in which the phrase structures are coarse-grained, but the lexicon and the POS tags are fine-grained. The result indicates that identifying the type of head-daughter dependencies is a hard task for a statistical parser, since the number of constituent labels in HPSG is higher than the labels in PSG. This might be the main reason that constituent parsers trained with PSG have higher performance than the ones trained with HPSG; consequently, the former parsers have a wider usage for NLP applications than the latter ones. However, it must be emphasized that coarse-grained annotation leads to lose valuable linguistic information that resembles the lexicon semantic information. Simplifying the parsing complexity by using coarse-grained phrase structures is useful for applications that require simple syntactic analyses of sentences; while the semantic information modeled in the HPSG-based data set is valuable for applications such as semantic role labeling which is a deep analysis.

In Model 5, both the POS tag and the phrase structure are coarse-grained and

only the lexicon is fine-grained to study the effect of the interaction between POS tag and phrase structure. As seen, there is a positive interaction between them which determines that simplifying the model and losing the detailed syntactic information have a counter effect on parsing. Comparing Models 5 and 8 indicates that reducing data sparsity results in a high performance. In Model 6, the lexicon and the phrase structure are coarse-grained, while the POS tag is fine-grained. This model is built to study the effect of available morpho-syntactic information in case there is a reduction on data sparsity without the effect of the HPSG-based annotation. The results of Models 2-4 infer that class-based parsing, the detailed morpho-syntactic information in the POS tags, and the coarse representation of the annotation at the phrasal level have positive impacts on parsing. The impact of these three variables are represented together in Model 6 which outperforms all the experimented models. In contrast, Model 3 which has an opposite configuration performs the worst. In Model 7, the lexicon and the POS tag are coarse-grained and the phrase structure is fine-grained to study the effect of the HPSG-based annotation without the impact of the morpho-syntactic information but with less data sparsity. Comparing Models 7 and 8 indicates the negative impact of the HPSG-based annotation on parsing, since it is a hard task for the parser to determine the type of dependencies when a coarse representation of the syntactic categories is available. While a better performance is obtained when a finer representation of the syntactic categories is available as determined in Models 2. Finally, in Model 8, the coarse-grained representations of the information at the three dimensions are studied. Comparing Models 1 and 8 indicates that better results are obtained when there is a coarse representation of linguistic knowledge, but higher results will be obtained when, similar to Model 6, a richer POS tag is used.

The other observation on Table 1 is studying the effect of each annotation dimension on all possible configurations. Comparing Models 2 and 1, Models 6 and 4, Models 7 and 3, and Models 8 and 5 show that the former models beat the latter ones which indicates that the class-based model always outperforms the word-based model disregarding the annotation of the POS tag and the phrase structure. There can be a similar study on the effect of POS tag annotation by comparing Models 1 and 3, Models 2 and 7, Models 4 and 5, and Models 6 and 8. All former models outperform the latter ones which indicates the superiority of fine-grained POS tag annotation disregarding the lexicon and phrase structure. To study the impact of phrase structure annotation, Models 4 and 1, Models 5 and 3, Models 6 and 2, and Models 8 and 7 are compared. All former models perform better than the latter ones which shows that the coarse-grained phrase structure annotation always results in a higher parsing performance disregarding the lexicon and POS tag. It has to be mentioned that the differences between the performance of all of the eight models are statistically significant according to the 2-tailed t-test ($p < 0.01$).

4 Conclusion

In this paper, we studied the effect of annotation granularity on parsing from three dimensions (lexicon, POS tag, and phrase structure) on Persian. Comparing the

results with the baseline determined that coarse-grained representation of lexicon has a positive impact on parsing. The detailed morpho-syntactic information of POS tags plays an important role on parsing and missing this information drops its performance. Determining the type of head-daughter dependencies is a hard task which reduces parsing performance.

5 Acknowledgement

Masood Ghayoomi is funded by the German research council DFG under the contract number MU 2822/3-1.

References

[1] M. Bijankhan. The role of corpora in writing grammar. *Journal of Linguistics*, 19(2):48–67, 2004. Tehran: Iran University Press.

[2] P.F. Brown, P.V. deSouza, R.L. Mercer, V.J.D. Pietra, and J.C. Lai. Class-based n-gram models of natural language. *Computational Linguistics*, 18:467–479, 1992.

[3] M. Candito and B. Crabbe. Improving generative statistical parsing with semi-supervised word clustering. In *Proc.s of the Int. Conf. on Parsing Technologies*, 2009.

[4] E. Charniak. A maximum-entropy-inspired parser. In *Proc.s of the 1st NAACL Conf.*, NAACL 2000, pages 132–139, 2000.

[5] M. Collins. *Head-Driven Statistical Models for Natural Language Parsing*. PhD thesis, University of Pennsylvania, 1999.

[6] M. Ghayoomi. Bootstrapping the development of an HPSG-based treebank for Persian. *Linguistic Issues in Language Technology*, 7(1), 2012.

[7] M. Ghayoomi. Word clustering for Persian statistical parsing. In *JapTAL '12: Proc.s of the 8th Int. Conf. on Advances in NLP*, pages 126–137, 2012.

[8] D. Klein and C.D. Manning. Accurate unlexicalized parsing. In *Proc.s of the 41st Meeting of the ACL*, pages 423–430, 2003.

[9] T. Koo, X. Carreras, and M. Collins. Simple semi-supervised dependency parsing. In *Proc.s of the ACL-08*, 2008.

[10] G. Leech and A. Wilson. *Standards for Tagsets*, pages 55–80. Text, speech, and language technology. Kluwer Academic Publishers, 9 edition, 1999.

[11] M.P. Marcus, B. Santorini, and M.A. Marcinkiewicz. Building a large annotated corpus of English: the Penn treebank. *Computational Linguistics*, 19(2), 1993.

[12] Y. Miyao. *From Linguistic Theory to Syntactic Analysis: Corpus-Oriented Grammar Development and Feature Forest Model*. PhD thesis, University of Tokyo, 2006.

[13] C.J. Pollard and I.A. Sag. *Head-Driven Phrase Structure Grammar*. University of Chicago Press, 1994.

[14] Ines Rehbein and Josef vanGenabith. Treebank annotation scheme and parser evaluation for German. In *Proc.s of EMNLP-CoNLL*, pages 630–639, 2007.

[15] K. Simov, P. Osenova, A. Simov, and M. Kouylekov. Design and implementation of the bulgarian HPSG-based treebank. *Research on Language and Computation*, 2:495–522, 2004.

[16] A. Stolcke. SRILM - An extensible language modeling toolkit. In *Proc.s of the Int. Conf. on Spoken Language Processing*, 2002.

Automatic Coreference Annotation in Basque

Iakes Goenaga, Olatz Arregi, Klara Ceberio,
Arantza Díaz de Ilarraza and Amane Jimeno

University of the Basque Country UPV/EHU

iakesg@gmail.com

Abstract

This paper presents a hybrid system for annotating nominal and pronominal coreferences by combining ML and rule-based methods. The system automatically annotates different types of coreferences; the results are then verified and corrected manually by linguists. The system provides automatically generated suggestions and a framework for easing the manual portion of the annotation process. This facilitates the creation of a broader annotated corpus, which can then be used to reiteratively improve our ML and rule-based techniques.

1 Introduction

Coreference resolution task is crucial in natural language processing applications like Information Extraction, Question Answering or Machine Translation. Machine learning techniques as well as rule-based systems have been shown to perform well at resolving this task. Though machine-learning methods tend to dominate, in the CoNLL-2011 Shared Task[1], the best results were obtained by a rule-based system (Stanford's Multi-Pass Sieve Coreference Resolution System [13]).

Supervised machine learning requires a large amount of training data, and the spread of machine learning approaches has been significantly aided by the public availability of annotated corpora produced by the 6th and 7th Message Understanding Conferences (MUC-6, 1995 and MUC-7, 1998) [17, 18], the ACE program [9], and the GNOME project [22]. In the case of minority and lesser-resourced languages, however, although the number of annotated corpora is increasing, the dearth of material continues to make applying these approaches difficult. Our aim is to improve this situation for Basque by both improving coreference resolution and facilitating the creation of a larger corpus for future work on similar tasks.

We will design a semi-automatic hybrid system to speed up corpus tagging by facilitating human annotation. Our system will allow the annotation tool to

[1] http://conll.cemantix.org/2011/task-description.html

display nominal and pronominal coreference chains in a user-friendly and easy-to-understand manner, so that coreferences can be tagged or corrected with a simple click of the mouse.

We will annotate, at coreference level, the Reference Corpus for the Processing of Basque (EPEC), a 300,000 word collection of written standard Basque that has been automatically tagged at different levels (morphology, surface syntax, and phrases). We will endeavor to solve nominal coreferences by combining rule-based techniques and machine learning approaches to pronominal anaphora resolution. The machine learning techniques will allow us to make use of existing resources for Basque, while using rule-based techniques for nominal coreference resolution will allow us to partially circumvent the problem of the limits of those resources.

Our machine learner is trained on a part of the Eus3LB Corpus[2] [15], a collection of previously parsed journalistic texts. This corpus, the basis of the EPEC corpus, has been manually tagged at coreference level, but only pronominal anaphora are annotated [1].

The paper is organized as follows. Section 2 describes some of the most significant work on coreferences, followed by a section presenting the tools we use to annotate corpora automatically and our aim in carrying out this work. In section 4 we describe the automatic coreference resolution process, which is divided into two parts: nominal coreference resolution process and pronominal anaphora resolution process. Section 5 then presents our experimental results and section 6 closes the paper with some concluding remarks.

2 Related Work

Recent work on coreference resolution has been largely dominated by machine learning approaches. In the SemEval-2010 task on Coreference Resolution in Multiple Languages[3] [24], most of the systems were based on those techniques [7, 26, 12]. Nevertheless, rule-based systems have also been applied successfully: in the CoNLL-2011 Shared Task, for example, the best result was obtained by [13], which proposes a coreference resolution system that is an incremental extension of the multi-pass sieve system proposed in [23]. This system is shifting from the supervised learning setting to an unsupervised setting.

At the same time, most other systems proposed at CoNLL-2011 [8, 6, 10] were based on machine learning techniques. The advantage of these approaches is that there are many open-source platforms for machine learning and machine learning based coreference such as BART[4] [27] or the Illinois Coreference Package [5].

The state of the art for languages other than English varies considerably. A rule-based system for anaphora resolution in Czech is proposed in [14], which uses Treebank data containing more than 45,000 coreference links in almost 50,000

[2] Eus3LB is part of the 3LB project.

[3] http://stel.ub.edu/semeval2010-coref/systems

[4] http://www.bart-coref.org/

manually annotated Czech sentences. Most recently, a substantial amount of newly annotated data for Czech has prompted the application of a supervised machine learning approach to resolving noun phrase coreferences in Czech [20]. On the other hand, [16] presents an approach to Persian pronoun resolution based on machine learning techniques. Other authors present an end-to-end coreference resolution rule-based system for Polish [21].

3 The Tagging Process

The annotation of coreference in Basque starts out with an annotated corpus that provides us with an easier work environment, one that focuses on the specific structures that could be part of a reference chain. The EPEC corpus has been morphosyntactically analyzed by means of MORFEUS [2]. After that, two automatic taggers (rule-based and stochastic) disambiguate at the lemmatization level. Finally, entities, chunks and complex postpositions are identified by means of the following tools: i) EIHERA, which identifies entities (Institution, Person and Location) [3]; and ii) IXATI Chunker [11], which identifies verb chains, noun phrase units, and complex postpositions.

Referring to the annotation of pronominal anaphora, 25.000 words of this corpus was carried out manually. For this tagging, we used the MMAX2 application [19] (adapted to the established requirements). Although the annotation tool is adequate, the process is still arduous and time consuming; we wanted to make it faster and more user-friendly. Toward this end, we developed an automatic coreference resolution system and transported the results it produced into the MMAX2 output window. Thus, depending on the type of coreference, the tool now displays either the possible chains or the possible antecedents. Coreference mentions appear highlighted in the text, so that simply by clicking on a coreference the annotator can see the chain of elements belonging to the same cluster. For each pronominal anaphor, the five most probable antecedents are linked, and the most probable is highlighted. The annotator needs only to choose the correct one.

4 The Coreference Resolution Process

The input of the coreference resolution module consists of a part of the EPEC corpus where each word of the corpus contains its form, lemma, category, POS and morphosyntactic features such as case and number. In addition, NPs are also labeled in the corpus. We only take into account NPs as potential mentions to be included in a coreference chain. The boundaries of these NPs are defined using three labels (*NP*, *NPB*, and *NPE*): if the NP contains more than a word, one label indicates the beginning [NPB] and the other one indicates the end [NPE]. Otherwise, if the NP contains only one word, the word is tagged with a unique label [NP] at its end.

Correct noun phrase tagging is crucial to coreference resolution: if a noun phrase is tagged incorrectly in the corpus, a potential anaphor or antecedent will be lost. We have detected 124 mislabeled noun phrases in our corpus, representing 9% of the total number of NPs. Most of these cases have a beginning label but no end label, an error that is due to the use of Constraint Grammar to annotate NPs automatically. This formalism does not verify if the beginning label has been annotated when it annotates an end label and vice versa. To avoid having this problem hamper our coreference task, we attempted to correct some of these mislabeled cases some simple rules, with varying success. These rules look for the opening and the ending label. Therefore, if one of them lacks, the heuristic tags the missing one. After the corrections were carried out, we started the coreference resolution process with 1265 correct noun phrases.

We divided our coreference resolution process into two subtasks depending on the type of coreference: nominal coreference resolution and pronominal coreference resolution. We used a rule-based method to solve nominal coreferences while employing a machine learning approach to find pronominal anaphora and their antecedents.

4.1 Nominal Coreference Resolution

We implemented the nominal coreference resolution process as a succession of three steps: (1) classifying noun phrases in different groups depending on their morphosyntactic features; (2) searching and linking coreferences between particular groups, thus creating possible coreference clusters; and (3) attempting to eliminate incorrect clusters and return correct ones by means of the coreference selector module, which takes into account the order of the noun phrases in the text.

4.1.1 Classification of Noun Phrases

In order to find easily simple coreference pairs, we classify noun phrases into seven different groups (G1...G7) according to their morphosyntactic features. Some of these features are proposed in [28]. To make an accurate classification, we create extra groups for the genitive constructions.

G1: The heads of those NPs that do **not** contain any **named entity** (see [23]). Although we use the head of the noun phrase to detect coreferences, the entire noun phrase has been taken into account for pairing or clustering purposes.

For example: *Escuderok [euskal **musika** tradizionala] eraberritu eta indartu zuen*. (Escudero renewed and gave prominence to [traditional Basque **music**]). The head of this NP is *musika* ("music"). Hence, we save this word in the group.

G2: NPs that contain **named entities with genitive form**. For example: ***[James Bonden** lehen autoa] Aston Martin DB5 izan zen*. ([**James Bond's** first car] was an Aston Martin DB5).

G3: NPs that contain **named entities with place genitive form**. For example: ***[Bilboko]** biztanleak birziklatze kontuekin oso konprometituak daude*. ([The

citizens **of Bilbao**] are very involved in recycling).

G4: NPs that contain **name entities with place genitive form and genitive form** (in Basque, the *-ko* and *-en* suffixes). In other words, this group contains NPs that fulfill the conditions for both G2 and G3. For example:

*[**Jesulinen Bilboko** etxea] Bilboko lekurik onenean dago.* ([**Jesulin's Bilbao** house] is located in the best area of Bilbao).

G5: NPs that contain **named entities**. These named entities can not have any form of genitive or place genitive. For example: *[**Leo Messi**] munduko futbol jokalaririk hoberena izango da segur aski*. ([**Leo Messi**] is probably the best football player in the world).

G6: Appositions + named entities. For example: *[**Pakito Mujika presoa**] orain dela gutxi epaitu dute*. ([**The prisoner Pakito Mujika**] has been judged recently).

G7: Postpositional phrases. Basque has a postpositional system (instead of prepositions, as in English), and therefore we mark the independent postposition and the preceding NP as a unit. For example: *Joan den astean [**Moriren aurka**] aurkeztutako zentsura mozioak krisia sortu zuen LDPn*. (The vote of no confidence [**against Mori**] caused a crisis in the LDP last week).

4.1.2 Candidate Selection

The aim of this module is to find the most obvious clusters of coreferences that will then be evaluated in the next module. To create these clusters we consider two main factors: (1) how to match mentions to decide if they are relevant candidates and (2) in which groups we must search for candidate mentions.

To decide whether two particular mentions are coreferential, we use two different matching techniques. Which one we select depends on the number of words in the mentions: if the number of words is the same in both cases, we use Exact Match, otherwise we use Relaxed Head Matching [23]. Let us explain these two mechanisms.

Exact Match (EM): To consider two mentions with the same number of words coreferential, they have to be equal character for character. In the case of the mentions of the no named entities group (G1), we use the mention heads for matching. For example: *[The **dog**] was huge and running free... [This **dog**] is the same one that bit John*.

Those two noun phrases belong to group 1 and their heads (dog) coincide character for character. So for the time being we consider them potential coreferences and save them, pending a final decision in the last module.

Relaxed Head Matching (RHM): We consider two mentions potential coreferences when all the words of the shortest mention are included in the longest mention with no alteration of order and with no intermediate words. For example, the system matches the mention *James Morrison* to the mention *James Morrison Strous* because all the words that compose the shortest mention are in the other mention without any alteration. But it does not match the mention *James Morri-*

	G1	G2	G3	G4	G5	G6	G7
G1	x	x	x	x		x	
G2	x	x					
G3	x		x				
G4	x			x			
G5					x	x	
G6	x				x	x	
G7							x

Table 1: Possible combinations

son to the mention *James Lewis Morrison*, because although all the words of the first mention are included in the second mention, the second mention contains an intermediate word (Lewis).

In order to decide in which groups to look for candidate mentions, we can use our knowledge of the composition of the groups. Thus, we know that coreferences between some groups are more likely than between others. For example, we can expect to find coreferences between the mentions of group 5 and group 6 because both groups include named entities, like the mentions *Brad Pitt* (G5) and *the actor Brad Pitt* (G6). By contrast, there will be no coreferences between the elements of the group 1 and group 5, for the simple reason that group 1 is created with no named entities and group 5 is created with named entities only. Consider the mention *carpenter* (G1) and the mention *John Carpenter* (G5). Although the word *carpenter* appears in both mentions, in the first one it refers to a profession and in the second to a surname. Therefore, we have discarded some combinations; the possible ones are summarized in Table 1.

Once each mention is filed in its proper group and it is clear what combinations of groups we need to look for possible coreferences in, we can begin searching for possible coreferences for each mention.

First, we search the mentions of each group for possible candidates. Consider a group with the mentions *Juan Montero, Perez, Montero, Gallastegi, Buesa*. The system would link the mentions *Juan Montero* and *Montero*. For some groups (G1 and G5), the system next tries to link mentions with mentions in other groups. For example, the mentions included in group 1 [*the dog*...] are the most general, so we match them with mentions from several groups (G2 [*George's dog*...], G3 [*the dog in Rome*...], G4 [*Nuria's Madrid dog*...], G6 [*the dog Larry*...]) due to the probability of finding coreferences in these morphologically compatible groups. However, it is also possible that two mentions in different groups could be coreferential only through a third mention. For example, we cannot directly join the mentions *Michael Jordan* and *basketball player* because we lack a clue that we could use to make the connection. But if we were to find the mention *Michael Jordan the basketball player*, we could use it to join all three mentions.

4.1.3 Coreference Selector Module

The objective of this module is to return clusters of coreferences, validating or modifying the clusters that it receives from the previous module. The input of this module, then, is a set of clusters that links coreference candidates.

In order to decide whether a cluster of coreferences is correct, the order in which the mentions of the cluster appear in the text is crucial. We can find the same mention in the text twice without it being coreferential. Consider this example: *[Iñaki Perurena] has lifted a 325-kilo stone... [Perurena] has trained hard to get this record... His son [Jon Perurena] has his father's strength... [Perurena] has lifted a 300-kilo cube-formed stone*. In this example we find the mention Perurena twice. The previous module links these mentions, creating a cluster of four mentions [*Iñaki Perurena, Perurena, Jon Perurena, Perurena*]. However, *Jon Perurena* and *Iñaki Perurena* are not coreferential, so this cluster is not valid. To eliminate this type of erroneous linkage, the coreference selector module takes into account the order in which the mentions appear. In other words, it matches the mention *Iñaki Perurena* to the first *Perurena* mention and the mention *Jon Perurena* to the second *Perurena* mention, as this is most likely to have been the writer's intention.

The system labels all marked mentions as possible coreferents (m_1, m_2, m_3, m_4, m_n) and then proceeds through them one by one trying to find coreferences. The system uses the following procedure. (1) If there exists a mention (for example m_3) that agrees with an earlier mention (for example m_1) and there **does not exist** a mention between them (for example m_2) that is coreferential with the current mention (m_3) and not with the earlier mention (m_1), the module considers them (m_1 and m_3) coreferential. (2) If there **exists** a mention between the two mentions (for example m_2) that is coreferential with the current mention (m_3) but not with the earlier mention (m_1), the module will mark as coreferents the intermediate mention (m_2) and the current mention (m_3). Thus, the module forms, step by step, different clusters of coreferences, and the set of those strings forms the final result of the Nominal Coreference Resolution Process.

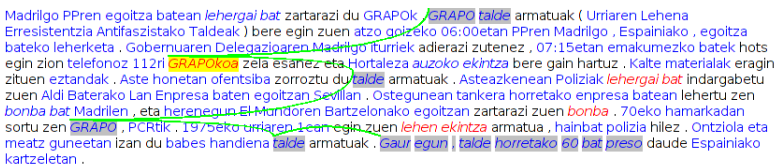


Figure 1: Nominal coreference example.

Figure 1 shows the result of the Nominal Coreference Resolution represented by the MMAX2 tool. The annotator then checks if the cluster of coreferences is correct. If it is, the coreference is annotated simply by clicking on it; if it is not, the annotator can easily correct it. Thus, the time employed in annotating coreferences is reduced.

4.2 Pronominal Anaphora Resolution

In order to use a machine learning method, a suitable annotated corpus is needed. As noted in the introduction, we use part of the Eus3LB Corpus. This corpus contains 349 annotated pronominal anaphora and it's different from the data we use to evaluate and develop our pronominal and nominal coreference resolution systems. The method used to create training instances is similar to the one explained in [25]. Positive instances are created for each annotated anaphor and its antecedents, while negative instances are created by pairing each annotated anaphor with each of the preceding noun phrases that are between the anaphor and the antecedent. Altogether, we have 968 instances; 349 of them are positive, and the rest (619) negative.

The method we use to create testing instances is the same we use to create training instances—with one exception—. As we can not know *a priori* what the antecedent of each pronoun is, we create instances for each possible anaphor (pronouns) and the eight noun phrases nearest to it. We choose the eight nearest NPs because experiments on our training set revealed that the antecedent lies at this distance 97% of the time. Therefore, for each anaphor we have eight candidate antecedents, i.e., eight instances. The features used are obtained from the linguistic processing system defined in [4].

The next step is to use a machine learning approach to determine the most probable antecedents for each anaphor. After consultation with linguists, we decided on returning a ranking of the five most probable antecedents. The most probable antecedent will be highlighted. Then, these clusters of coreferences are displayed in the MMAX2 tool. In this manner, the annotator will select the correct antecedent for each anaphor from a set of five possible antecedents, instead of having to find it in the whole text, saving time and improving performance. Consequently, we will be able to create a larger tagged corpus faster as well as achieve better models for applying and improving the machine learning approach.

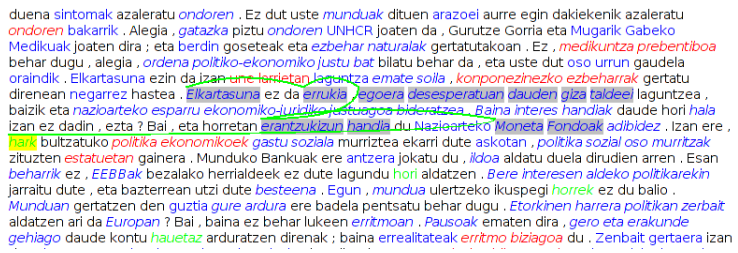


Figure 2: Pronominal coreference example.

Figure 2 represents an anaphor (*hark*, in English *he/she/it*) and its five most probable antecedents in the MMAX2 window. The annotator can choose the correct antecedent of the pronominal anaphor with a few clicks.

Nominal	P	R	F1
MUC	75.33%	81.33%	78.21%
B^3	72.80%	83.95%	77.97%
BLANC	90.5%	87.57%	88.98%
Pronominal	76.9%	60.0%	67.4%

Table 2: Results of the anaphora resolution system.

5 Experimental Results

We use two different strategies to evaluate the two coreference resolution processes, since we use two different methods to link coreferences. In the pronominal anaphora resolution process, we return the five most probable antecedents for each anaphor, while in the nominal coreference resolution process we return a cluster of mentions that links coreferential mentions for each nominal coreference.

The evaluation metrics are chosen with a view toward appropriateness. We use the classic measures (precision, recall and F1) to evaluate the pronominal anaphora resolution process, counting as success the instances when the real antecedent of the pronominal anaphor is among the five most probable antecedents. To evaluate nominal coreferences, we use BLANC, MUC and B^3 metrics, as they are the three most significant metrics used for this task.

We use 1004 NPs to develop our nominal coreference resolution system and 281 NPs to evaluate it. For the evaluation of our pronominal anaphora resolution system, we use 130 pronominal anaphora of those 1285 NPs.

We present the results of our coreference resolution system in Table 2. In the nominal coreference resolution system we obtain an F-score of at least 78% using the three above-mentioned metrics. On the other hand, using the pronominal coreference resolution system, the F-score is 67.4%. Although these results are not the best obtained in coreference resolution systems, they build a solid base for improving our system and indicate that our system is of considerable use in speeding up the manual nominal/pronominal anaphora annotation. This, in turn, will allow us to create a broader corpus and use it to improve our hybrid approach to automatic corpus annotation.

6 Conclusions and Future Work

In this work we present a system for automatically annotating nominal and pronominal coreferences using a combination of rules and ML methods. Our work begins by detecting incorrectly tagged NPs and, in most cases, correcting them, recovering 63% of the incorrectly tagged NPs. Next, in the case of the nominal coreferences, we divide the NPs into different groups according to their morphological features to find coreferences among the compatible groups. Then we use a ML approach to solve pronominal anaphora; this returns, for each anaphor, a cluster that contains

the anaphor and its five most probable antecedents.

Our results demonstrate that, despite their simplicity, ML and deterministic models for coreference resolution obtain competitive results. This will allow us to create an automatic annotation system to improve the manual annotation process of the corpora. A larger tagged corpus, in turn, will enable us to improve the performance of our automatic system.

Acknowledgements

This work has been supported by Hibrido Sint (TIN2010-20218) project.

References

[1] I. Aduriz, K. Ceberio, and A. Díaz de Ilarraza. Pronominal anaphora in Basque: Annotation issues for later computational treatment. In *DAARC2007 Lagos (Portugal); pp. 1-7, ISBN: 978-989-95343-0-8*, 2007.

[2] I. Alegria, X. Artola, K. Sarasola, and M. Urkia. Automatic morphological analysis of Basque. 1996.

[3] I. Alegria, N. Ezeiza, and I. Fernandez. Named entities translation based on comparable corpora. In *Multi-Word-Expressions in a Multilingual Context Workshop on EACL06. April 3. pp.1-8. Trento*, 2006.

[4] O. Arregi, K. Ceberio, A. Díaz de Ilarraza, I. Goenaga, B. Sierra, and A Zelaia. A first machine learning approach to pronominal anaphora resolution in Basque. In *IBERAMIA 2010. LNAI 6433*, 2010.

[5] E. Bengtson and D. Roth. Understanding the value of features for coreference resolution. In *In EMNLP*, 2008.

[6] A. Björkelund and P. Nugues. Exploring lexicalized features for coreference resolution. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, CONLL Shared Task'11, pages 45–50, Stroudsburg, PA, USA, 2011.

[7] S. Broscheit, M. Poesio, S. Paolo Ponzetto, K. J. Rodriguez, L. Romano, O. Uryupina, Y. Versley, and R. Zanoli. BART: A multilingual anaphora resolution system. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval'10, pages 104–107, Stroudsburg, PA, USA, 2010.

[8] K. Chang, R. Samdani, A. Rozovskaya, N. Rizzolo, M. Sammons, and D. Roth. Inference protocols for coreference resolution. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, CONLL Shared Task '11, pages 40–44, Stroudsburg, PA, USA, 2011.

[9] G. Doddington, A. Mitchell, M. Przybocki, L. Ramshaw, S. Strassel, and R. Weischedel. The Automatic Content Extraction (ACE) Program–Tasks, Data, and Evaluation. *Proceedings of LREC 2004*, pages 837–840, 2004.

[10] C. N. dos Santos and D. L. Carvalho. Rule and tree ensembles for unrestricted coreference resolution. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, CONLL Shared Task'11, pages 51–55, Stroudsburg, PA, USA, 2011.

[11] J. M. Arriola A. Atutxa A. Diaz-De-Ilarraza N. Ezeiza K. Gojenola M. Oronoz A. Soroa R. Urizar I. Aduriz, M. J. Aranzabe. Methodology and steps towards the construction of EPEC, a corpus of written Basque tagged at morphological and syntactic levels for the automatic processing. pages 1–15. Rodopi. Book series: Language and Computers, 2006.

[12] H. Kobdani and H. Schütze. Sucre: A modular system for coreference resolution. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval'10, pages 92–95, Stroudsburg, PA, USA, 2010.

[13] H. Lee, Y. Peirsman, A. Chang, N. Chambers, M. Surdeanu, and D. Jurafsky. Stanford's Multi-Pass Sieve Coreference Resolution System at the CoNLL-2011 Shared Task. In *Proceedings of the CoNLL-2011 Shared Task*, 2011.

[14] N. G. Linh. Rule-based approach to pronominal anaphora resolution applied on the Prague dependency treebank 2.0 data, 2007.

[15] A. Díaz de Ilarraza L. Moreno E. Bisbal-M. J. Aranzabe A. Ageno M. A. Marti F. Navarro M. S. Palomar, M. Civit. Construcción de una base de datos de arboles sintáctico-semánticos para el catalán, euskera y español. In *XX. Congreso SEPLN*, 2004.

[16] N. S. Moosavi and G. Ghassem-Sani. A ranking approach to Persian pronoun resolution. Advances in Computational Linguistics. In *Advances in Computational Linguistics. Research in Computing Science 41*, pages 169–180, 2009.

[17] MUC-6. Coreference task definition (v2.3, 8 sep 95). In *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pages 335–344, Columbia, Maryland, USA, November 6–8 1995.

[18] MUC-7. Coreference task definition (v3.0, 13 jul 97). In *Proceedings of the 7th Message Understanding Conference (MUC-7)*, Fairfax, Virginia, USA, April 29–May 1 1998.

[19] C. Müller and M. Strube. Multi-level annotation of linguistic data with MMAX2. In Sabine Braun, Kurt Kohn, and Joybrato Mukherjee, editors, *Corpus Technology and Language Pedagogy: New Resources, New Tools, New Methods*, pages 197–214. Peter Lang, Frankfurt a.M., Germany, 2006.

[20] M. Novák and Z. Žabokrtský. Resolving noun phrase coreference in Czech. In *Proceedings of the 8th international conference on Anaphora Processing and Applications*, DAARC'11, pages 24–34, Berlin, Heidelberg, 2011. Springer-Verlag.

[21] M. Ogrodniczuk and M. Kopeć. End-to-end coreference resolution baseline system for Polish, 2011.

[22] M. Poesio. The mate/gnome proposals for anaphoric annotation, revisited. In *In Michael Strube and Candy Sidner (editors), Proceedings of the 5th SIGdial Workshop on Discourse and Dialogue*, pages 154–162, 2004.

[23] K. Raghunathan, H. Lee, S. Rangarajan, N. Chambers, M. Surdeanu, D. Jurafsky, and C. D. Manning. A multi-pass sieve for coreference resolution. In *EMNLP*, pages 492–501, 2010.

[24] M. Recasens, L. Màrquez, E. Sapena, M. A. Martí, M. Taulé, V. Hoste, M. Poesio, and Y. Versley. Semeval-2010 task 1: Coreference resolution in multiple languages. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval'10, pages 1–8, Stroudsburg, PA, USA, 2010.

[25] W. M. Soon, D. Chung, Y. Lim, and H. T. Ng. A machine learning approach to coreference resolution of noun phrases, 2001.

[26] O. Uryupina. Corry: A system for coreference resolution. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval'10, pages 100–103, Stroudsburg, PA, USA, 2010.

[27] Y. Versley, S. P. Ponzetto, M. Poesio, V. Eidelman, A. Jern, J. Smith, X. Yang, and A. Moschitti. Bart: a modular toolkit for coreference resolution. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Demo Session*, HLT-Demonstrations '08, pages 9–12, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[28] R. Vieira and M. Poesio. An empirically based system for processing definite descriptions. *Comput. Linguist.*, 26(4):539–593, December 2000.

Analyzing the Most Common Errors in the Discourse Annotation of the Prague Dependency Treebank

Pavlína Jínová, Jiří Mírovský, Lucie Poláková
Charles University in Prague
Institute of Formal and Applied Linguistics
Malostranské nám. 25, Prague 1, Czech Republic
E-mail: (jinova|mirovsky|polakova)@ufal.mff.cuni.cz

Abstract

We present an analysis of the inter-annotator discrepancies of the Czech discourse annotation in the Prague Dependency Treebank 2.0. Having finished the annotation of the inter-sentential semantic discourse relations with explicit connectives in the treebank, we report now on the results of the evaluation of the parallel (double) annotations, which is an important step in the process of checking the quality of the data. After we shortly describe the annotation and the method of the inter-annotator agreement measurement, we present the results of the measurement and, most importantly, we classify and analyze the most common types of annotators' disagreement.

1 Discourse in the Prague Dependency Treebank 2.0

The Prague Dependency Treebank 2.0 (PDT; Hajič et al. [3]) is a manually annotated corpus of Czech journalistic texts, annotated on three layers of language description: morphological, analytical (the surface syntactic structure), and tectogrammatical (the deep syntactic structure) (Hajič et al. [2]). On the tectogrammatical layer, the data consist of almost 50,000 sentences.

Annotation of discourse structure in PDT uses a lexical approach, similarly to one of the Penn Discourse Treebank (PDTB 2.0, Prasad et al. [9]). It is based on identifying a discourse connective (an expression with text structuring function), which takes two text segments as its arguments and indicates a discourse meaning between them. The annotators mark three basic types of information: the connective (contrary to the Penn approach, there is no list of possible discourse connectives in advance), the two arguments of the connective (mostly clauses, sentences but sometimes also larger units, such as paragraphs) and the semantic type of the relation. At this stage of the project, we do not annotate implicit relations (relations without a connective).

A set of discourse semantic types was developed as a result of comparison of the sense hierarchy used in Penn (Miltsakaki et al. [5]) and the set of Prague tectogrammatical labels called functors (Mikulová et al. [4]).

Annotators had at their disposal both plain text and the tectogrammatical analysis (tree structures). The annotation was carried out on the tectogrammatical trees; a specialized annotation tool was developed for this purpose (Mírovský et al. [7]). The process of annotation spanned over two years (Mladová et al. [8]). Altogether in all PDT data, there have been 8,834 inter-sentential discourse arrows annotated.

Example I shows two sentences with a discourse relation of type *opposition* between them:

(I) *1. Čtyři ostrovní státy nabídly 260 vojáků.*
[Four island states offered 260 soldiers.]
*2. Podle mluvčího Pentagonu jich **ale** budou zapotřebí aspoň dva tisíce.*
*[**However**, according to the Pentagon spokesman, at least two thousand of them will be needed.]*

2 Evaluation of parallel annotations

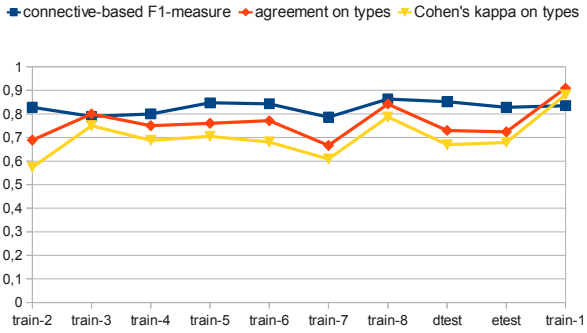
Several annotators annotated the data but (for obvious reasons of limited resources) each part of the data has only been annotated by one of them. Only 4% of the data (2,084 sentences) have been annotated in parallel by two annotators. We used the parallel (double) annotations for measuring the inter-annotator agreement, and for analyzing the most common errors, i.e. difficult parts of the annotation. Altogether, there have been 44 documents, 2,084 sentences and 33,987 words annotated in parallel.

To evaluate the inter-annotator agreement (IAA) on selected texts annotated in parallel by two annotators, we used the connective-based F1-measure (Mírovský et al. [6]), a simple ratio, and Cohen's κ (Cohen [1]). The connective based F1-measure was used for measuring the agreement on the recognition of discourse relations, a simple ratio and Cohen's κ were used for measuring the agreement on the type of relations recognized by both the annotators.[1]

In the connective-based measure, we consider the annotators to be in agreement on recognizing a discourse relation if two connectives they mark (each of the connectives marked by one of the annotators) have a non-empty intersection (technically, a connective is a set of tree nodes). For example, if one of the annotators marks two words *a proto [and therefore]* as a connective, and the other annotator only marks the (same) word *proto [therefore]*, we take it as agreement – they both recognized the presence of a discourse relation. (They still may disagree on its type.)

1 In all our measurements, only inter-sentential discourse relations have been counted.

Graph 1 shows results of subsequent measurements of the agreement between the two most productive annotators during the two years of annotation. Each measurement was taken on approx. 200 sentences (3 to 5 documents).



Graph 1: The inter-annotator agreement in the subsequent measurements

The overall F1 measure on the recognition of discourse relations was 0.83, the agreement on types was 0.77, and Cohen's κ was 0.71. Altogether, one of the annotators marked 385 inter-sentential discourse relations, the other one marked 315 inter-sentential discourse relations.

The simple ratio agreement on types (0.77 on all parallel data) is the closest measure to the way of measuring the inter-annotator agreement used on subsenses in the annotation of discourse relations in the Penn Discourse Treebank 2.0, reported in Prasad et al. [9]. Their agreement was 0.8.

Table 1 shows a contingency table of the agreement on the four major semantic classes, counted on the cases where the annotators recognized the same discourse relation. The simple ratio agreement on the four semantic classes is 0.89, Cohen's κ is 0.82. The agreement on this general level in the Penn Discourse Treebank 2.0 was 0.94 (Prasad et al. [9]).

	contr	**contin**	**expans**	**tempor**	**total**
contr	137	2	5	1	145
contin	1	49	5		55
expans	4	8	60	3	75
tempor		1	1	7	9
total	142	60	71	11	284

Table 1: A contingency table of the agreement on the four discourse super types (semantic classes)

3 Analysis of the discrepancies

In our measurements and analyses of the IAA, we observe two main types of disagreement: (1) disagreement in identifying the connective, i.e. a situation when one annotator recognized a connective that the other did not, and (2) disagreement in the semantic type of the relation, i.e. a situation when both the annotators recognized a relation anchored by the same connective but they did not characterize this relation in the same way semantically.

3.1 Disagreement in identifying the discourse connective

The analysis of the cases of disagreement in the discourse connective identification shows that it is in the vast majority a mistake of one of the annotators (80 cases in sum). A situation where different interpretations of both annotators are correct only occurs once in our parallel data.

10% of these disagreement cases are rather of a technical than of a linguistic nature: both annotators marked the discourse relation but one of them forgot to add the connective. 15% represent the cases in which one of the annotators overlooked a typical connective – and with it also the relation itself, 75% stand for cases in which the connective is represented by an expression that in Czech also has a non-connective function. This ambiguity probably contributes to the fact that these expressions are more easily left out when reading the text than the typical connectives are. This applies especially for so-called rhematizers, i.e. particles with a rheme signalling function (e.g. *také [also]*, *jenom [only]*, for details on rhematizers see Mikulová et al. [4]).

3.2 Disagreement in the semantic type of a discourse relation

Two main types of disagreement can be distinguished in determining the semantic type of a discourse relation (approximately 60 cases in sum):

First, there are cases of disagreement that are clearly a mistake of one of the annotators: the context does not allow for the asserted interpretation. It represents 26% of the total amount of disagreement and we see their main source in misunderstanding the text.

Second, some cases of disagreement in the semantic type cannot be considered mistakes of the annotators. 7% arise as a consequence of the fact that some of the relations seemed to be defined very clearly but in the course of the annotation they proved to be quite difficult to distinguish from one another in a complicated real-data situation (e.g. a case of relation of *explication* vs. relations of *reason-result* and *specification*).

The remaining cases of disagreement in the semantic type can be divided into (i) agreement and (ii) disagreement within the four major semantic classes. (i) represents situations where each annotator assigned a different type of a relation **within one** of the four basic semantic classes (which are *temporal relations***,** *contingency*, *contrast* and *expansion*) and both these interpretations are equally correct (approx. 26% of all cases of disagreement

in the semantic type). This situation is typical especially for relations from the *contrast* group. This type of disagreement does not need to be treated as a complete disagreement: the IAA measurement method in the Penn Discourse Treebank 2.0 considers such cases as agreement on the higher level in the sense hierarchy (cf. Table 1).

The remaining type of disagreement are cases where two semantic types **from different** major semantic classes have been used for interpretation of one relation (approx. 41% of all cases of the disagreement in type). These cases of disagreement arise directly from contexts that allow both interpretations. Often, some semantically rather vague connectives contribute to those cases. It is illustrated by example (II). The relation between the sentences in the example text was interpreted as *reason-result* (one argument is the reason of the other one) and also as *conjunction* (the second argument adds new information to the first one). Both interpretations are possible here.

(II) *Za nabídku by se nemusel stydět ani Don Carleone - nebylo možné jí odolat. **A tak** do roka a do dne dostalo práci 440 shanonských občanů a do pěti let jich bylo už desetkrát tolik.*

*Not even Don Carleone would have to be ashamed of that offer - it was impossible to resist. **And so** 440 people of Shannon got a job within a year and a day, and within five years, they were already ten times as many.*

This type of disagreement between a relation from the *contingency* group (e.g. *reason-result*) and a relation from the *expansion* group (e.g. *conjunction, equivalence*) is the most frequent disagreement across different major semantic classes. This situation, in our opinion, follows from a certain grade of vagueness of some journalistic formulations – we are allowed to treat the text sequences both causally and as a simple build-up of the previous context.

4 Conclusion

We presented an evaluation and analysis of disagreements in the annotation of the inter-sentential discourse relations with an explicit connective in the Prague Dependency Treebank. The results show that agreeing on an existence of a discourse relation via a surface-present discourse connective is a manageable task, whereas the annotation of the semantic type of the relation depends heavily on the interpretation of the text. The comparison of our annotation with the annotation of Penn Discourse Treebank 2.0 showed that on two levels of granularity of discourse types (senses), the inter-annotator agreement was roughly the same in these two projects.

Almost all cases of disagreement in identifying a connective have to be interpreted as a mistake of an annotator. A majority of them originates from the fact that some of these expressions in Czech have also other functions than the one of a discourse connective. As for the disagreement in the

semantic type, annotators' mistakes are not so frequent. This type of disagreement arises from (1) semantic closeness of some relation types, (2) semantic ambiguity/vagueness of some contexts, and (3) in the case of the relation of *explication* also from the complex nature of the relation.

Acknowledgments

We gratefully acknowledge support from the Grant Agency of the Czech Republic (grants P406/12/0658 and P406/2010/0875) and the Ministry of Education, Youth and Sports in the Czech Republic, program KONTAKT (ME10018) and the LINDAT-Clarin project (LM2010013).

References

[1] Cohen J. (1960) A coefficient of agreement for nominal scales. *Educational and Psychological Measurement, 20(1)*, pp. 37–46

[2] Hajič J., Vidová Hladká B., Böhmová A., Hajičová E. (2000) The Prague Dependency Treebank: A Three-Level Annotation Scenario. In: *Treebanks: Building and Using Syntactically Annotated Corpora (ed. Anne Abeille)*, Kluver Academic Publishers

[3] Hajič J., Panevová J., Hajičová E., Sgall P., Pajas P., Štěpánek J., Havelka J., Mikulová M., Žabokrtský Z., Ševčíková-Razímová M. (2006) Prague Dependency Treebank 2.0. *Software prototype, Linguistic Data Consortium,* Philadelphia, PA, USA, ISBN 1-58563-370-4, www.ldc.upenn.edu

[4] Mikulová M. et al. (2005) Annotation on the tectogrammatical layer in the Prague Dependency Treebank. *Annotation manual.* Available from http://ufal.mff.cuni.cz/pdt2.0/doc/manuals/en/t-layer/pdf/t-man-en.pdf

[5] Miltsakaki E., Robaldo L., Lee A., Joshi A. (2008) Sense annotation in the Penn Discourse Treebank. In: *Computational Linguistics and Intelligent Text Processing, Lecture Notes in Computer Science, Vol 4919,* pp. 275–286

[6] Mírovský J., Mladová L., Zikánová Š. (2010) Connective-Based Measuring of the Inter-Annotator Agreement in the Annotation of Discourse in PDT. In: *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010),* Beijing, China, pp. 775–781

[7] Mírovský J., Mladová L., Žabokrtský Z.: Annotation Tool for Discourse in PDT. In: *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, Tsinghua University Press, Beijing, China, pp. 9–12

[8] Mladová L., Zikánová Š., Hajičová E. (2008) From Sentence to Discourse: Building an Annotation Scheme for Discourse Based on Prague Dependency Treebank. In: *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008),* Marrakech, Morocco

[9] Prasad R., Dinesh N., Lee A., Miltsakaki E., Robaldo L., Joshi A., Webber B. (2008) The Penn Discourse Treebank 2.0. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*, CD-ROM

Will a Parser Overtake Achilles? First experiments on parsing the Ancient Greek Dependency Treebank

Francesco Mambrini∗ and Marco Passarotti†
∗University of Cologne (Germany)
†Università Cattolica del Sacro Cuore, Milan (Italy)

Abstract

We present a number of experiments on parsing the Ancient Greek Dependency Treebank (AGDT), i.e. the largest syntactically annotated corpus of Ancient Greek currently available (350k words ca). Although the AGDT is rather unbalanced and far from being representative of all genres and periods of Ancient Greek, no attempt has been made so far to perform automatic dependency parsing of Ancient Greek texts. By testing and evaluating one probabilistic dependency parser (MaltParser), we focus on how to improve the parsing accuracy and how to customize a feature model that fits the distinctive properties of Ancient Greek syntax. Also, we prove the impact of genre and author diversity on parsing performances.

1 Introduction

Among the languages currently spoken in the world, Greek has one of the longest documented histories. The first texts written in Greek that have survived to our days date from the mid of the second millennium BCE (around 1420-1400). From the phase that is commonly known as Ancient Greek (9th Century BCE - 6th Century CE), a vast literature has been preserved and thoroughly studied. In rough numbers, two of the most important digital collection of Ancient Greek texts, which are far from being exhaustive, the Thesaurus Linguae Graecae (TLG)[1] and the Perseus Digital Library[2], contain respectively more than 105 and 13 million words.

1.1 Annotated corpora of Ancient Greek

Some of the Ancient Greek texts are today included in two different treebanks.

The Ancient Greek Dependency Treebank (AGDT) [3] is a dependency-based treebank of literary works of the Archaic and Classical age published by Perseus[3].

[1] TLG: `http://www.tlg.uci.edu/`, which goes so far as to the fall of Byzantium (1453 BCE).
[2] Perseus Digital Library: `http://www.perseus.tufts.edu/hopper/`
[3] AGDT: `http://nlp.perseus.tufts.edu/syntax/treebank/greek.html`

In its theoretical framework and guidelines, the AGDT is inspired by the analytical layer of annotation of the Prague Dependency Treebank of Czech [5]. Currently, the last published version of AGDT (1.6) includes 346,813 tokens.

The collection is constituted by unabridged works only: four major poets (Homer, Hesiod, Aeschylus, Sophocles)[4], belonging to two literary genres (epic poetry – Homer, Hesiod – and tragedy – Aeschylus and Sophocles), are represented, as well as one single work of philosophical prose (the *Euthyphro* by Plato). Chronologically, the texts range from the 8th to the late 5th Century BCE. The composition of the AGDT 1.6 is resumed in table 1.

Author	Work	Tokens
Aeschylus	Agamemnon	9,806
	Eumenides	6,380
	Libation Bearers	6,563
	Prometheus Bound	7,064
	Seven Against Thebes	6,206
	Suppliants	5,949
Hesiod	Shield of Heracles	3,834
	Theogony	8,106
	Works and Days	6,941
Homer	Iliad	128,102
	Odyssey	104,467
Sophocles	Ajax	9,474
	Women of Trachis	8,811
	Electra	10,458
	Antigone	8,716
	Oedipus King	9,746
Plato	Euthyphro	6,097
Total		346,813

Table 1: AGDT 1.6: Composition

The Pragmatic Resources in Old Indo-European Languages corpus (PROIEL) [6], on the other hand, is a multilingual parallel corpus of translations of the New Testament in a selection of Indo-European languages; the Greek section includes also other prose texts of different periods (four books of Herodotus' *Histories*, 5th Century BCE, and Palladius' *Historia Lausiaca*, 5th Century CE). The syntactic

[4]For Hesiod and Aeschylus, the AGDT includes the *opera omnia* of the integrally preserved works (fragments are excluded). Of Sophocles' 7 extant tragedies, 5 are annotated. A tradition that dates from the Antiquity and was followed by convention in the AGDT indicates the legendary figure of Homer as the author of *Iliad* and *Odyssey* (along with other minor compositions); the real existence of one single author for both poems has been denied by the modern Homeric scholarship.

annotation is also based on a dependency grammar, and it is partially inspired by the AGDT. The total of the Ancient Greek annotated data is presently 153,730 tokens[5].

1.2 Open questions and methodology

So far, the annotation of both treebanks has been performed manually. The two collections started by annotating some of the most important texts for current university curricula in Classics. Both on account of the difficulty of the works, which require a hard linguistic and philological training, and of the pedagogical value that can be attached to manual annotation, no use of NLP techniques has been made so far. With the partial exception of [7][6], no comprehensive study has been dedicated to evaluate and improve the performance of parsers on Ancient Greek.

Yet, the available syntactically annotated corpora cover only a small portion of the attested documents, in terms of quantity as well as of representativeness of the different genres and chronological phases. The vast majority of the prose production is not included in the AGDT. Moreover, even for the two genres that are adequately represented, a substantial number of texts still remain to be annotated; apart from the missing 2 tragedies of Sophocles, this is the case with the 19 plays of Euripides (170,118 words, 5th Century BCE) or, for epic poetry, the *Argonautica* of Apollonius Rhodius (45,478 words, 3rd Century BCE) or the so-called *Homeric Hymns* (18,211 words, traditionally attributed to Homer, ranging chronologically from the 7th Century BCE to Late Antiquity).

An efficient dependency parser for Ancient Greek is thus a major acquisition supporting the creation of a balanced annotated corpus.

A previous study on Latin treebanks [12], which share a number of features with our collections, has shown that genre and chronology have a decisive influence on the accuracy of different parsers. Three questions then appear to be relevant for a preliminary study:

1. what are the parser's settings ("feature model") that fit best the AGDT?
2. what is the impact of genre and author on the parser's performances?
3. following 2, how should a training set be built? Is the size more relevant than data homogeneity in terms of genre and author?

In this paper, we provide a first answer to these three questions by studying the performances of one single dependency parser, which is trained and tested on different sections of our corpus. From the top-ranking list of the CoNLL-X shared task[7], we selected MaltParser [10], on account of its flexibility, in terms of

[5] This total was communicated to us by the general editor of the project. As the annotation process is still in progress, the numbers change every day.

[6] By leveraging a Greek-Hebrew parallel corpus, [7] develops a target-specific method to improve the parsing accuracy of the Greek Old Testament.

[7] `http://ilk.uvt.nl/conll/`

both parsing algorithm and feature setting. For the task of algorithm and feature selection, we used MaltOptimizer [2] as a starting point, whose output we have evaluated and further improved[8].

1.3 The data

Our data were taken from the AGDT and converted to the CoNLL format[9].

The texts taken from the AGDT were organized by author and genre, in order to evaluate how MaltParser performs with varying authors and genres. It is well known [11] that non-projectivity crucially affects the efficiency of dependency parsers. In comparison with the treebanks used in CoNLL-X [4, 155, tab. 1] and CoNLL 2007 shared tasks [9, 920, tab. 1], our data show a remarkably higher rate of non-projective arcs.

The subsets that we used, along with the number and percentage of non-projective arcs, are resumed in table 2[10].

Data set	Works/Authors	Sentences	Tokens	Non-proj. arcs	%
Homer	Il., Od.	15175	232569	62013	26.66
Tragedy	Aesch., Soph.	7897	95363	21747	22.80
Sophocles	Soph.	3873	47205	10456	22.15

Table 2: Data sets

Each of the subsets in table 2 was randomly partitioned into 5 training and testing sets, all in a ratio of approximately 9:1, in order to perform 5 different experiments.

We first focused on the test sets of Homer (table 3) and Sophocles (table 4).

Then, we studied how genre and author affect MaltParser in detail. We used a model trained on the Homeric poems to evaluate 3 different subsets: (a) the whole annotated work of Hesiod (18,881 tokens: same genre as the training set, but different author), (b) a sample from Sophocles of roughly the same size as Hesiod (18,418 tokens: different genre and author), and (c) the whole available Plato (6,091 tokens: different genre, different author, prose text).

[8] In all our experiments, we used LIBSVN as learning algorithm for MaltParser.

[9] The CoNLL format includes the following 10 fields, although only the first 8 contain non-dummy values: ID (token counter), FORM, LEMMA, CPOSTAG (coarse-grained PoS), POSTAG (fine-grained PoS), FEATS (unordered set of morphological features), HEAD (head of current token, i.e. a value of ID), DEPREL (dependency relation to the HEAD); `http://nextens.uvt.nl/depparse-wiki/DataFormat`

[10] Since not all of the tragedies of Sophocles were already published at the time when we started our work, we used an unfinished version of the *Oedipus King*: 1361 tokens from the ca. last 100 lines of the play were unannotated.

Set Name	Sentences	Tokens	% Train/Test
Homer_Test1	1686	25898	11.14
Homer_Test2	1562	23258	10.00
Homer_Test3	1469	23267	10.00
Homer_Test4	1500	23274	10.01
Homer_Test5	1486	23259	10.00

Table 3: Homer: Test sets

Set Name	Sentences	Tokens	%Train/Test
Sophocles_Test1	430	5186	10.99
Sophocles_Test2	389	4725	10.01
Sophocles_Test3	389	4726	10.01
Sophocles_Test4	384	4731	10.02
Sophocles_Test5	386	4721	10.00

Table 4: Sophocles: Test sets

2 Results and evaluation

2.1 Algorithm and feature selection

Not surprisingly, given the above reported non-projective rates in our data sets, MaltParser scores rather poorly with the default algorithm (Nivre, a linear-time algorithm limited to projective dependency structures) and model (Arceager). With this configuration, training and testing the parser on the Homeric poems (tab. 3), we attained the following results (baseline): 44.1% LAS, 60.3% UAS, 49.2% LA [4][11].

By applying the options suggested by MaltOptimizer, we increased the accuracy of the parser considerably. Due to the high number of non-projective trees and of nodes attached to the root, MaltOptimizer recommends the adoption of the following options[12]:

- adoption of a non-projective algorithm: Covington non-projective is suggested;
- use of the label "AuxK" (terminal punctuation) as default for unattached

[11] The metrics used are the following: Labeled Attachment Score (LAS): the percentage of tokens with correct head and relation label; Unlabeled Attachment Score (UAS): the percentage of tokens with the correct head; Label Accuracy (LA): the percentage of tokens with the correct relation label.

[12] For a quick introduction to MaltParser optimization, see [8]; for a detailed explanation of each option see [1].

tokens that are attached to the technical root node;

- covered root set to "left" (see [8, 7]);
- "shift" allowed: the parser is allowed to skip remaining tokens before the next target token is shifted to the top of the stack;
- root node treated as a token: root dependents are allowed to be attached with a RightArc transition.

The feature model suggested by MaltOptimizer is reported in Appendix, tab. 10.

Starting from these customized options for the Covington non-projective algorithm, we modified the feature model, according to our knowledge of both Ancient Greek and the annotation style of AGDT. We tested and evaluated 6 different configurations: the best performances were attained with experiment n. 4 (Exp4).

The feature model was then modified according to the requirements of the other non-projective algorithm (Stacklazy), so as to evaluate the parser with both non-projective algorithms available for MaltParser[13]. Then, we compared the performances using two different training and test sets: Homer (trained on Homer) and Sophocles (trained on the Tragedy training set: see sec. 2.2 for the choice). Table 5 lists the results of these experiments with MaltOptimizer configuration (MO) and with our feature models for both algorithms (Exp4), all compared to the baseline (first line of tab. 5).

Test set	Training	Algorithm	Feature mod.	LAS	UAS	LA
Homer	Homer	nivreager	arceager	44.1	60.3	49.2
Homer	Homer	covnonproj	MO	69.02	76.46	78.66
Homer	Homer	covnonproj	Exp4	70.96	77.9	80.34
Homer	Homer	stacklazy	Exp4	71.72	78.26	81.62
Soph.	Tragedy	covnonproj	MO	55.24	64.12	67.48
Soph.	Tragedy	covnonproj	Exp4	57.7	65.52	70.14
Soph.	Tragedy	stacklazy	Exp4	56	63.92	69.12

Table 5: Evaluation of algorithms and models: Average of 5 experiments

The configuration Exp4 improves the accuracy for all the metrics. Covington's algorithm surpasses Stacklazy with the corpus of the tragic poems, but the opposite is true in the case of *Iliad* and *Odyssey*.

We first evaluated the accuracy of the parser by grouping the results of the best scoring configuration (Exp4) by dependency relations (tab. 6). The main depen-

[13]The modified feature models are reported in the Appendix, tab. 11 (Covington) and tab. 12 (Stacklazy).

dency relations (PRED, OBJ, SBJ, ADV, ATR, PNOM)[14] attain a good level of accuracy in both subsets, while the performances decrease considerably whenever coordination is concerned.

	Homer		Sophocles	
DepRel	Cov	St	Cov	St
ADV	77.48	75.12	58.94	55.78
ADV_CO	39.32	38.92	18.56	7.36
ATR	75.52	74.84	62.38	64.96
ATR_CO	39.04	39.88	16.28	12.86
AuxC	63.48	57.78	47.96	41.56
AuxP	77.7	73.72	58.72	57.82
OBJ	79.44	76.08	61.26	59.36
OBJ_CO	57.98	63.18	27.36	28.56
PNOM	69.44	68.46	38.28	32.04
PNOM_CO	36.48	36.68	1.82	1.82
PRED	83.52	81.56	85.6	74.8
PRED_CO	61.7	78.78	32.42	43.78
SBJ	82.96	81.5	64.74	58.86
SBJ_CO	58.48	61.26	11.64	6.1

Table 6: Accuracy (LAS) grouped by DEPREL

Non-projectivity's impact on results is quite strong. As reported in table 7, accuracy is considerably lower in case of non-projective relations.

Finally, we grouped the results by PoS tag. Fig. 1 show the different performances for each of the 13 part-of-speech labels used in the AGDT.

[14]The tag PRED is given to the predicate of the main clause of a sentence. An ATR is a sentence member that further specifies a noun in some respect; typical attributives are adjectives, relative clauses and nouns in the genitive case. The difference between OBJ and ADV roughly corresponds to the one between arguments and adjuncts of verbs or adjectives. SBJ: subject. PNOM: nominal predicate. AuxP: prepositions; AuxC: conjunctions. In the event that a node is member of a coordinated construction, the DEPREL label is appended with the suffix _Co.

	Homer		Sophocles	
Projectivity	Cov	St	Cov	St
Projective	72.18	73.32	60.28	58.34
Non-proj.	60.84	58.46	36.24	36.5

Table 7: Accuracy (LAS) grouped by arc projectivity

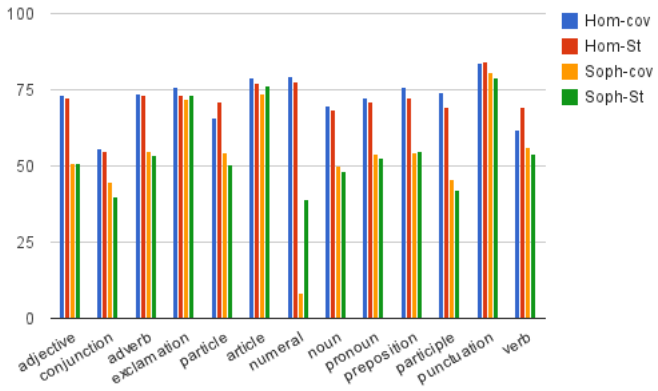


Figure 1: Accuracy (LAS) grouped by PoS tag

2.2 Differences in genre and author

In order to evaluate the role of genre and author diversity and the composition of the best training set both in terms of accuracy of the results and of computational costs, we performed two different experiments.

Firstly, we parsed the 5 sets of approximately 5,000 tokens of Sophocles (tab. 4) using our best performing configuration (Exp4 with Covington non-projective) and with models trained on: (a) the Sophocles training set, (b) all the available tragedies, and (c) the whole AGDT. The average results are reported in table 8.

Training	LAS	UAS	LA	Learning Time
Sophocles	56.14	64.32	68.82	09:20
Tragedy	57.7	65.52	70.14	42:15
AGDT	57.65	66	70	14:26:42

Table 8: Accuracy (LAS) for Soph. with different training sets

As it can be seen, adding more data from texts belonging to the same genre (i.e. tragedy) improves the parser's performances sensibly. Training a model on the whole available treebank slightly decreases the accuracy of LAS and LA, and furthermore, at the cost of a disproportionate growth of learning time. It seems that, in order to parse a text of an author like Sophocles, building a training set from works of the same genre is the most rewarding strategy.

The importance of genre homogeneity is confirmed also by the second experiment we performed. Using our best performing feature models (Exp4 for Coving-

ton and Stacklazy), we trained a model on the available epic poems of Homer and we used it to parse the whole work of the epic poet Hesiod, a sample of roughly equivalent size taken from the tragedies of Sophocles and the only philosophic dialogue of Plato that is included in the AGDT. The results are reported in table 9.

	Cov			Stack		
Test set	LAS	UAS	LA	LAS	UAS	LA
Hesiod	60.7	69.3	72	60.9	68.8	73.7
Sophocles	48.48	58.72	61.3	46.84	56	61.24
Plato	47.1	60.7	60.1	48.2	60.2	62.9

Table 9: Different authors with models trained on Homer

Hesiod's text, the one which is closer to the language of the poems used in the training set both chronologically and for genre, performs far better than the two others. Plato's set is considerably smaller, but it seems that a prose text like a philosophical dialogue is more difficult to parse for a parser trained on epic poetry than a complex poetic text like the one of Sophocles. This result confirms the well known fact that the Homeric poems are a fundamental model and a source of inspiration for the language of Greek poetry. Further studies will be required as soon as new prose texts are added to the collection, in order to confirm this result.

3 Conclusion and future work

We have focused on the performances of a probabilistic dependency parser (MaltParser) on Ancient Greek literary texts. By tuning a series of features, we have considerably improved the efficiency of the parser.

In the process, we have used MaltOptimizer and further improved the feature model suggested by the software, by using the lemmata of the words (in addition to forms) and introducing other modifications that are resumed in tab. 11 and 12.

From our experiments, it emerged that, in order to parse a given Ancient Greek literary work, texts that belong to the same genre as the target text should be used as a training set, rather than the totality of the available collection. This is proved in the case of Sophocles, whose work is not yet fully annotated. Our results can help in providing a more accurate and reliable basis for semi-automatic annotation of the remaining texts of this author and, arguably, for Euripides' work as well.

In the future, we will test other parsers, in order to compare their performances and see whether we can further improve our preliminary results. In particular, we intend to test: DeSR, ISBN, MST, Anna (Mate-Tools)[15]. We will also include

[15]DeSR: `https://sites.google.com/site/desrparser/`; ISBN: `http://cui.unige.ch/~titov/idp/`; MST: `http://www.seas.upenn.edu/~strctlrn/MSTParser/MSTParser.html`; Anna: `http://code.google.com/p/mate-tools/`.

prose texts from the PROIEL corpus, in order to improve the performances on prose texts[16].

Appendix: Feature Models

The feature models used for the experiments are reported in the following tables. The lines of the tables are grouped by Feature Function and Column Name. For an explanation of Feature Functions, Address Functions (both parsing-algorithm-specific functions and dependency-graph functions), as well as of the syntax of the feature-model files, see [1].

Values of FEATS are split by "|" in Left[0], Left[1] for MO; in Left[0], Left[1], Right[0] for Exp4-Cov; in Stack[0], Stack[1], Stack[2], Stack[3] for Exp4-Stack.

Feature Function	Column Name	Address Function
InputColumn	FEATS	Left[0]; Right[0]
InputColumn	FORM	Left[0]; Right[0]; Right[1]; head(Left[0])
InputColumn	POSTAG	LeftContext[0] ;Left[0]; Left[1]; RightContext[0]; Right[0]; Right[1]; Right[2]; Right[3]
OutputColum	DEPREL	Left[0]; Right[0]; ldep(Left[0]); ldep(Left[0]); ldep(Right[0]); rdep(Left[0])

Table 10: MO feature model

References

[1] MaltParser user guide. http://www.maltparser.org/userguide.html.

[2] M. Ballesteros and J. Nivre. MaltOptimizer: a system for MaltParser optimization. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC 12)*, Istanbul, 2012. European Language Resources Association (ELRA).

[16]We would like to thank J. Nivre and M. Ballestreros for their kind support with Malt and MaltOptimizer. Professor R. Förtsch and R. Krempel (CoDArchLab, Cologne) also provided decisive help with the infrastructures of the laboratory for digital archaeology at the University of Cologne.

Feature Function	Column Name	Address Function
InputColumn	FEATS	Left[0]; Left[1]; Right[0].
InputColumn	FORM	LeftContext[0]; LeftContext[1]; LeftContext[2]; Left[0]; Left[1]; Left[2]; RightContext[0]; Right[0].
InputColumn	LEMMA	LeftContext[0]; LeftContext[1]; LeftContext[2]; LeftContext[3]; Left[0]; Left[1]; Left[2]; Left[3]; RightContext[0]; RightContext[1]; RightContext[2]; Right[0]; Right[1]; Right[2]; Right[3]; head(Left[0]).
InputColumn	POSTAG	LeftContext[0]; LeftContext[1]; LeftContext[2]; LeftContext[3]; Left[0]; Left[1]; Left[2]; Left[3]; RightContext[0]; RightContext[1]; RightContext[2]; Right[0]; Right[1]; Right[2]; Right[3]; head(Left[0]),
OutputColumn	DEPREL	Left[0]; Right[0]; ldep(Left[0]); ldep(Right[0]); rdep(Left[0]).

Table 11: Exp4-Cov feature model

Feature Function	Column Name	Address Function
InptuColumn	FEATS	Stack[0]; Stack[1].
InptuColumn	FORMS	Input[0]; Lookahead[0]; Stack[0]; Stack[1]
InputColumn	LEMMA	Input[0]; Lookahead[0]; Lookahead[1]; Lookahead[2]; Lookahead[3]; Stack[0]; Stack[1]; Stack[2].
InputColumn	POSTAG	Input[0]; Lookahead[0]; Lookahead[1]; Lookahead[2]; Lookahead[3].
InputColumn	POSTAG	Stack[0]; Stack[1]; Stack[2].
OutputColumn	DEPREL	Stack[0]; Stack[1]; ldep(Stack[0]); ldep(Stack[1]); rdep(Stack[0]); rdep(Stack[1]).

Table 12: Exp4-Stack feature model

[3] D. Bamman, F. Mambrini, and G. Crane. An ownership model of annotation: The Ancient Greek Dependency Treebank. In *Proceedings of the Eighth International Workshop on Treebanks and Linguistic Theories (TLT 8)*, pages 5–15, Milan, 2009. EDUCatt.

[4] S. Buchholz and E. Marsi. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X '06)*, CoNLL-X '06, pages 149–164, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

[5] A. Böhmová, J. Hajič, E. Hajičová, and B. Hladká. The Prague Dependency Treebank: A three-level annotation scenario. In A. Abeillé, editor, *Treebanks: Building and Using Syntactically Annotated Corpora*, pages 103–127. Kluwer, Boston, 2001.

[6] D. Haug and M.L. Jøhndal. Creating a parallel treebank of the old Indo-European Bible translations. In *Proceedings of the Second Workshop on Language Technology for Cultural Heritage Data (LaTeCH 2008)*, pages 27–34, Marrakech, 2008. European Language Resources Association (ELRA).

[7] J. Lee. Dependency parsing using prosody markers from a parallel text. In *Proceedings of the Ninth International Workshop on Treebanks and Linguistic Theories*, pages 127–39, Tartu, 2010. Northern European Association for Language Technology (NEALT).

[8] J. Nivre and J. Hall. Quick guide to MaltParser optimization. http://www.maltparser.org/guides/opt/quick-opt.pdf.

[9] J. Nivre, J. Hall, S. Kübler, R. McDonald, J. Nilsson, S. Riedel, and D. Yuret. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, Prague, Czech Republic, 2007. Association for Computational Linguistics.

[10] J. Nivre, J. Hall, and J. Nilsson. Maltparser: A data-driven parser-generator for dependency parsing. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC)*, pages 2216–2219, Genoa, 2006. European Language Resources Association (ELRA).

[11] J. Nivre and J. Nilsson. Pseudo-projective dependency parsing. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 99–106, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[12] M. Passarotti and P. Ruffolo. Parsing the *Index Thomisticus* Treebank. Some preliminary results. In P. Anreiter and M. Kienpointner, editors, *Latin Linguistics Today. Akten des 15. Internationalen Kolloquiums zur Lateinischen Linguistik, Innsbruck, 4. -9. April 2009*, volume 137, pages 714–725, Innsbruck, 2010. Institut für Sprachwissenschaft der Universität Innsbruck.

Using Parallel Treebanks for Machine Translation Evaluation

Magdalena Plamadă, Martin Volk

Institute of Computational Linguistics
University of Zurich
{plamada, volk}@cl.uzh.ch

Abstract

This paper presents a new method to evaluate machine translation (MT) systems against a parallel treebank. This approach examines specific linguistic phenomena rather than the overall performance of the system. We show that the evaluation accuracy can be increased by using word alignments extracted from a parallel treebank. We compare the performance of our statistical MT system with two other competitive systems with respect to a set of problematic linguistic structures for translation between German and French.

1 Introduction

An important step in improving the performance of a statistical machine translation (SMT) system is the diagnosis of its output. As human evaluation is expensive and the automatic metrics fail to convey information about the nature of the errors, researchers in the field have worked on linguistically-informed evaluation measures. The advantage of this approach is that it can pinpoint the weaknesses of MT systems in terms of morpho-syntactic errors.

Liu and Gildea [7] were among the first to incorporate syntactic features (and dependency relations) into MT evaluation. Their approach correlated better with the human judgments than the previous n-gram based metrics (e. g. BLEU) and has thus underlain following research in the field. Furthermore, semantic-based evaluation metrics such as [8] were developed with the purpose of assessing the meaning similarity. Latest approaches describe an evaluation metric which aims at incorporating several levels of linguistic information (lexical, morphological, syntactical and semantical) [3].

Although these metrics reflect various linguistic levels, they cannot perform a real diagnosis of MT systems. We therefore need a thorough analysis focused on different linguistic levels. In this paper, however, we only refer to the diagnosis of morpho-syntactic errors. Popović and Ney [11] proposed a method for identifying and analyzing translation errors involving different Part-of-Speech (PoS) classes.

Zhou et al. [15] introduced the idea of diagnostic evaluation based on linguistic checkpoints (see section 3) and released it as a stand-alone tool: Woodpecker[1]. Unfortunately, their tool works only for English-Chinese and is released under a restrictive license. On the other hand, a freely-available software, DELiC4MT[2], offers the same functionalities plus the option of adapting it to any language pair.

This paper builds upon previous research on linguistic checkpoints. Since this type of evaluation involves a fine-grained analysis of the texts in the source and target language, word correspondence is a very important prerequisite. Moreover, the quality of the evaluation strongly depends on the accuracy of these alignments. As both approaches use automatic alignment methods, the accuracy of the resulting alignments decreases. Therefore we suggest to avoid this drawback by extracting good alignments from a manually-checked parallel treebank.

This paper is structured as follows: In section 2 we describe our data and in the subsequent one the evaluation process. Section 4 introduces the changes we have made to the existing evaluation workflow. Section 5 presents and analyzes our experimental efforts. Finally, section 6 wraps up the discussion.

2 Our Reference Corpus: The Alpine Parallel Treebank

The reported experiments have been carried out on the German-French parallel treebank part of the SMULTRON corpus[3]. The treebank consists of 1000 sentences from the Text+Berg corpus[4], which contains the digitized publications of the Swiss Alpine Club from 1864 until 2011. The parallel treebank contains the "same" text in German and French, with most texts being translated from German into French and only a few of them vice versa.

We refer to a treebank as to a particular kind of annotated corpus where each sentence is mapped to a graph (a tree) which represents its syntactic structure. In addition to the syntactic annotation, the parallel treebank is aligned on the sub-sentential level, for example on the word or the phrase level. We regard phrase alignment as alignment between linguistically motivated phrases and not just arbitrary consecutive word sequences, as in statistical machine translation.

The annotation is a semi-automatic process, as we have manually checked and corrected the annotations at each processing step. PoS tagging is performed by the TreeTagger[5], with its standard parameter files for German and our in-house trained parameters for French, respectively. The tagged texts are then loaded into Annotate[6], a treebank editor which suggests constituent phrases and function labels based, in German, on the structures provided by the TnT Chunker[7]. For French, the

[1] http://research.microsoft.com/en-us/downloads/ad240799-a9a7-4a14-a556-d6a7c7919b4a
[2] http://www.computing.dcu.ie/~atoral/delic4mt/
[3] http://www.cl.uzh.ch/research/paralleltreebanks/smultron_en.html
[4] http://www.textberg.ch
[5] http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger
[6] http://www.coli.uni-saarland.de/projects/sfb378/negra-corpus/annotate.html
[7] http://www.coli.uni-saarland.de/~thorsten/tnt/

phrases are generated by a shallow parsing model (hidden Markov model) trained on the Le Monde corpus [1]. Finally, the monolingual treebanks are exported in the TIGER-XML format [6]. More details about the annotation of the German treebank can be found in [14], whereas the French annotation is described in [5].

We use the Bleualign algorithm [13] to align the sentences across both monolingual treebanks. Our alignment convention was to discard the automatic many-to-many word alignments for the purpose of increasing the precision. Subsequently, a human annotator checked and, when required, corrected the remaining word and sentence alignments and then added the phrase alignments. Finally, the alignment file is available in XML format, as the following snippet shows:

```
<align type="good" last_change="2010-09-03">
      <node treebank_id="de" node_id="s225_18"/>
      <node treebank_id="fr" node_id="s231_16"/>
</align>
```

This says that node 18 in sentence 225 of the German treebank (de) is aligned with node 16 in sentence 231 of the French treebank (fr). The node identifiers refer to the IDs in the TIGER-XML treebanks. The alignment is labeled as *good* when the linked phrases represent exact translations and as *fuzzy* in case of approximate correspondences.

3 The Evaluation Tool: DELiC4MT

DELiC4MT is an open-source tool that performs diagnostic evaluation of MT systems over user-defined linguistically-motivated constructions, also called *checkpoints*. This term was introduced by Zhou et al. [15] and refers to either lexical elements or grammatical constructions, such as ambiguous words, noun phrases, verb-object collocations etc. The experiments reported in this paper follow the workflow proposed by Naskar et al. [9], due to its option to integrate new language pairs.

First the texts are PoS-tagged and exported in the KYOTO Annotation Format (KAF)[2]. This scheme facilitates the inspection of the terms in the sentences and thus querying for specific features, such as PoS sequences. Figure 1 depicts the KAF annotation of the German phrase *den ersten Gipfel* (EN: the first peak) and its French equivalent *le premier sommet*.

The linguistic checkpoints are subsequently defined in the so-called kybot profiles. A kybot profile starts with the declaration of the involved variables and the relations among them and ends specifying which attributes of the matched terms should be exported. For example, figure 2 depicts the kybot profile for a nominal group consisting of a determiner, an adjective and a noun. Moreover, the constituent terms have to be consecutive. Once defined, the kybot profile is run over the source language KAF files and the matched terms (with the specified attributes) are stored in a separate XML file.

```
<text>[...]
  <wf wid="w729_6" sent="729" para="1">den</wf>
  <wf wid="w729_7" sent="729" para="1">ersten</wf>
  <wf wid="w729_8" sent="729" para="1">Gipfel</wf>
[...]</text>
<terms>[...]
 <term tid="t729_6" type="open" lemma="d" pos="ART">
  <span> <target id="w729_6"/> </span>
 </term>
 <term tid="t729_7" type="open" lemma="erst" pos="ADJA">
  <span> <target id="w729_7"/> </span>
 </term>
 <term tid="t729_8" type="open" lemma="Gipfel" pos="NN">
  <span> <target id="w729_8"/> </span>
 </term>
[...]</terms>

<text>[...]
  <wf wid="w729_6" sent="729" para="1">le</wf>
  <wf wid="w729_7" sent="729" para="1">premier</wf>
  <wf wid="w729_8" sent="729" para="1">sommet</wf>
[...]</text>
<terms>[...]
 <term tid="t729_6" type="open" lemma="le" pos="DET:ART">
  <span> <target id="w729_6"/> </span>
 </term>
 <term tid="t729_7" type="open" lemma="premier" pos="NUM">
  <span> <target id="w729_7"/> </span>
 </term>
 <term tid="t729_8" type="open" lemma="sommet" pos="NOM">
  <span> <target id="w729_8"/> </span>
 </term>
[...]</terms>
```

Figure 1: Sample KAF annotation for a German-French sentence pair

The last step evaluates how well did the MT system translate the linguistic phenomena of interest. The evaluation is based on n-gram similarity, thus counting the overlapping word sequences between the hypothesis (automatic translation) and the reference (the previously identified checkpoint instances). The evaluation module requires as input the source and target language texts in KAF format, as well as the word alignments between them, the XML file produced at the previous step and the automatic translation to be evaluated. Each matched instance is evaluated separately and, on this basis, the final score for the MT system is being computed. Figure 3 presents the evaluation of the noun phrases in figure 1. In this case, the hypothesis translation contains all the possible n-grams identified in the reference (6 variants for the 3-word phrase *le premier sommet*), so the instance receives the maximum score (6/6).

```
<Kybot id="kybot_a_n_de">
 <variables>
  <var name="X" type="term" pos="ART" />
  <var name="Y" type="term" pos="ADJ*" />
  <var name="Z" type="term" pos="NN*" />
 </variables>
 <relations>
  <root span="X" />
  <rel span="Y" pivot="X" direction="following" immediate="true" />
  <rel span="Z" pivot="Y" direction="following" immediate="true" />
 </relations>
 <events>
  <event eid="" target="$X/@tid" lemma="$X/@lemma" pos="$X/@pos"/>
  <role rid="" event="" target="$Y/@tid" lemma="$Y/@lemma" pos="$Y/@pos"
                                                        rtype="follows"/>
  <role rid="" event="" target="$Z/@tid" lemma="$Z/@lemma" pos="$Z/@pos"
                                                        rtype="follows"/>
 </events>
</Kybot>
```

Figure 2: A Kybot profile for a nominal group

4 MT Evaluation Method

Our extension with respect to the original version of the tool refers to the usage of alignments from the Alpine treebank. Previous papers on the topic [15, 9] had mentioned the limitation of automatically computed alignments and suggested methods to overcome the alignment noise, but none could compete the accuracy of a hand-aligned corpus. Therefore the strength of our approach consists in the integration of manually checked word alignments in the evaluation pipeline.

This required a special preprocessing step of extracting word alignments from the treebank and converting them to the format used by DELiC4MT. As an illustration, figure 4 depicts an aligned sentence pair from our treebank. Green lines indicate exact alignments and red lines represent fuzzy alignments. The corresponding alignments (from German to French) in the DELiC4MT format are:

```
0-0 1-2 2-3 3-4 4-5 5-6 5-7 6-8 6-9 6-10 6-11 7-12 8-14 8-15 ...
```

This means that the first word in the source sentence is aligned to the first one in the target sentence, taking into consideration that word numbering starts from 0.

The advantage of using a treebank over a word-aligned corpus is that the treebank contains other alignment types than “simple“ 1-1 word alignments. This often happens in German due to its frequent compounds, which are then represented as 1-n word alignments. For instance, the German compound *Montblanc-Abenteuer* (EN: Mont Blanc adventure) in figure 4 is aligned to the French noun phrase *aventure au Mont Blanc*. This correspondence is translated into the following word alignments: `6-8 6-9 6-10 6-11`.

```
Sen_id: 729 token_id: 6, 7, 8
Source tokens:  den, ersten, Gipfel
Alignments:  5-5, 6-6, 7-7,
Target equivalent ids: 5, 6, 7
Target sentence:
Après six heures j' atteignais le premier sommet, le Combin de la Tsessette.
Target equivalent tokens:  le premier sommet

Checking for n-gram matches for checkpoint instance: 319
Ref: le premier sommet
Hypo:
après six heures nous atteignons le premier sommet , le combin de la tsessette.
Number of 1-grams in reference: 3
# of matching 1-grams = 3
Number of 2-grams in reference: 2
# of matching 2-grams = 2
Number of 3-grams in reference: 1
Matched 3-gram: le premier sommet
# of matching 3-grams = 1
All n_gram matches :
le
...
le premier
premier sommet
le premier sommet

Total n_gram matches: 6
Total n_gram count in reference: 6
```

Figure 3: Sample output for a specific checkpoint instance

There are cases where a single word can correspond to a whole subtree in the other language. For example, the German adjective constituting the adjectival phrase *glücklich* (EN: happy) is paraphrased in French by the prepositional phrase *avec un sentiment de bonheur* (EN: with a feeling of happiness). We can thus use the phrase alignment in our treeebank and transform it into word alignments between the constituent words. In this way, our additional alignment level facilitates the extraction of n-m word alignments.

In order to demonstrate our claim, we have automatically computed the alignments for the 1000 sentences in the treebank with a well-established tool, GIZA++ [10]. Because the test set is relatively small for a statistical aligner to suggest accurate results, we have appended it to a considerably bigger corpus. For comparison purposes, we have chosen 200000 sentences from the Europarl corpus and, respectively, the same amount of sentences from an Alpine corpus. We have then used the generated alignments as input for the evaluation tool, along with the automatic translations generated by our SMT system (see section 5). Table 1 shows the results for several checkpoints for the language pair German-French.

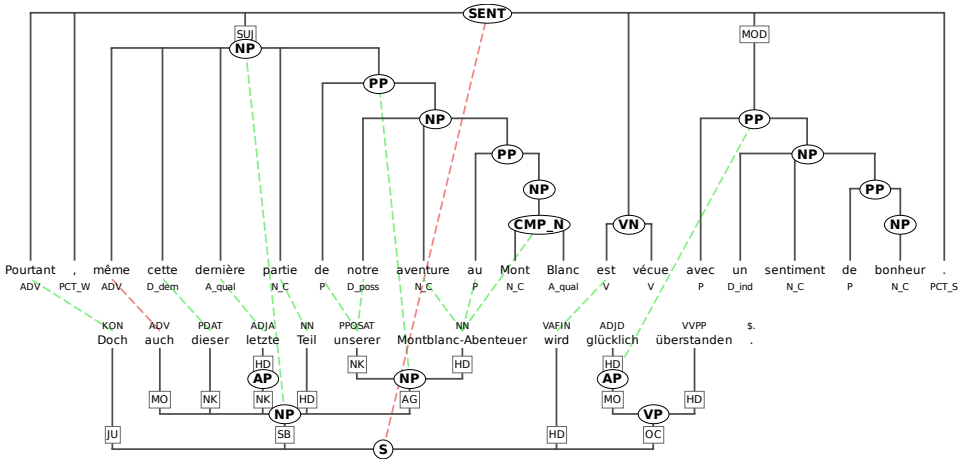


Figure 4: Aligned French-German tree pair from the Alpine treebank

The scores normally indicate the percentage of overlapping n-grams between the reference phrase (checkpoint instance) and the output produced by the MT system. However, in this context, the scores reported for the automatic alignments do not reflect the quality of the MT system. The evaluation module takes the same input in all three cases, except for the alignments, which are computed in different ways and generate different outcomes accordingly. Therefore the scores should be seen as estimates of the accuracy of the evaluation. The more precise the alignments, the more reliable the evaluation results.

We notice that the domain of the texts used for training GIZA++ does not influence significantly the accuracy, since the produced scores are similar (e.g. less than 2% difference between Europarl and the Alpine texts). However, when we compare the evaluation results with automatic alignments to the ones obtained with manual alignments, the latter ones are significantly better (up to 12% increase). This finding demonstrates the validity of our claim, namely that feeding manually proofed alignments from a parallel treebank to the evaluation pipeline generates more reliable results.

Checkpoint	**Alignment type**	**Final score**
Verb	GIZA++: Europarl	0.190 29
	GIZA++: Alpine	0.191 78
	Parallel Treebank	0.283 65
Det+Noun+Adj	GIZA++: Europarl	0.228 82
	GIZA++: Alpine	0.240 99
	Parallel Treebank	0.480 17

Table 1: Evaluation results for different alignments

Checkpoint	Instances	Google	PT	Our system
Noun	4790	0.313 72	0.351 29	**0.418 57**
Verb	953	0.211 94	**0.307 69**	0.283 65
Det_Adj_N	278	0.379 28	0.445 72	**0.480 17**
Dass_Pron_Verb	20	0.375 37	**0.383 56**	**0.383 56**
Verb_Pron_DetNoun	17	0.224 97	0.244 09	**0.409 45**
Weil_Pron_Verb	10	0.236 26	0.311 11	**0.577 78**
Pass	7	0.134 50	0	**1**

Table 2: Evaluation results for German-French

5 Evaluation Experiments

In this experiment, we compare our in-house SMT system with 2 other systems, Google Translate[8] and Personal Translator (PT)[9], in terms of handling specific linguistic checkpoints. Our SMT system was trained according to the instructions for building a baseline system at WMT 2011[10], with the difference that we use MGIZA++ [4] for computing the alignments. As training data we use Alpine texts from the Text+Berg corpus (approx. 200000 sentence pairs German-French).

The test corpus comprises 1000 sentence pairs from our Alpine treebank. For all systems, we use the manually-checked alignments extracted from the treebank. The comparison will be based on checkpoints which we considered particularly interesting for each translation direction, most of them PoS-based.

Table 2 contains the evaluation results for the language pair German-French. We have investigated the following checkpoints: nouns, finite verbs, noun phrases consisting of a determiner, an adjective and a noun (Det_Adj_N), subordinate clauses introduced by *dass* (EN: that) and *weil* (EN: because) and verb-subject-object collocations (Verb_Pron_DetNoun). Additionally, we have also considered the ambiguous word *Pass* (EN: passport, mountain pass, amble).

One notices that Personal Translator usually performs better than Google, probably because, being a rule-based system, it is aware of grammatical constructions and knows how to handle them properly. Its weaknesses are mostly related to the choice of words and unknown words, respectively. Since we are now looking at particular grammatical structures, it is likely for a rule-based system to analyze them adequately. Another evidence for this claim is the fact that Personal Translator outperforms all the other systems with respect to finite verbs, which pose difficulties in German (e. g. separable verbs).

Our in-house MT system performs in all cases better than its opponents because it has been trained with texts from the same domain. It thus gains strongly in vocabulary coverage. The most striking example is the German word *Pass* (EN:

[8] http://translate.google.com

[9] http://www.linguatec.net/products/tr/pt

[10] http://www.statmt.org/wmt11/baseline.html

Checkpoint	Instances	Google	PT	Our system
Noun_de_Noun	346	0.225 09	0.183 07	**0.415 81**
Poss_Noun	289	0.292 73	0.390 96	**0.469 55**
que_V	224	0.292 99	0.292 99	**0.340 76**
que_Pr_V	115	0.390 41	0.397 26	**0.445 21**
Ne_V_Pas	45	0.447 37	0.381 58	**0.526 32**
Verb_Pr_Vinf	25	0.158 73	0.301 59	**0.396 83**
V_Vinf_DetN	23	0.133 93	**0.294 64**	0.250 00

Table 3: Evaluation results for French-German

mountain pass), translated as *col* in the mountaineering domain, but as either *passe-port* (EN: passport) or *la passe* (EN: pass [the ball]) in other contexts. The analysis of the words with lemma *Pass* is reproduced in the last line of table 2. We notice that Personal Translator could not reproduce the correct meaning of the word in this context. Google succeeded getting one checkpoint instance right due to the collocation with a proper noun. As a result, it could correctly translate *Forcola-Pass* as *col Forcola*.

For the opposite translation direction, we have chosen linguistic structures particular for the source language (French): nominal phrases consisting of two nouns separated by the preposition *de* (such as in *journée d'été*) or of a possessive adjective and a noun (such as in *notre voisin*). Besides, we have selected relative clauses introduced by the word *que*, which can be interpreted as either a relative pronoun or a conjunction. Finally, we have considered verbal constructions: the negative form of verbs (e.g. *ne résisterait pas*), modal constructions involving nouns (e.g. *doit affronter le parcours*) and pronouns (e.g. *peut nous aider*), respectively. The results are presented in table 3.

The best-handled construction by all three systems is the negative verbal form, followed by the relative clause introduced by *que* and followed by a pronoun. If we remove the restriction that *que* is directly followed by a pronoun, the particle becomes ambiguous and this causes the drop of 10% between the scores in the third and the fourth line. Noun phrases are also handled well by our system, whereas complex verbal phrases raise challenges. The rule-based system Personal Translator gets the best score in the latter case due to its linguistic knowledge background, as we have noticed before.

5.1 Limitations of the System

As DELiC4MT evaluation uses string-based comparisons, it penalizes every small difference from the reference text. Consequently it will equally penalize a MT system for dropping a word as for misspelling a single letter from a word. This is particularly disadvantageous for SMT systems, which use no linguistic information (grammar, syntax or semantics). On the other hand, some of the limitations of

string-based comparisons can be easily overcome by considering not only word forms, but also lemmas or synsets. In the following, we outline some types of word form variations, which resulted in penalized errors:

- **Singular/Plural inconsistencies**: The tool distinguishes between singular and plural forms, although the word stem is the same. In the example below, the German sentence uses the singular form of the noun *Spur*, which is then translated in French as *la trace*. However, the reference translation suggests the plural form *les traces* in the given context, so the sentence pair is counted as a failed checkpoint, although the translation is fairly good.

  DE: *Im Abendrot bewundern wir die Spur unserer mutigen Vorgänger.*
  Automatic translation: *Au soleil couchant, nous pouvons admirer la trace de nos courageux prédécesseurs.*
  FR Reference: *Au coucher du soleil, nous admirons les traces de nos courageux prédécesseurs.*

- **Verbal tense inconsistencies**: If the MT system expresses the verbal phrase in a slightly different way, the tool will penalize the difference. For example, the finite verb *bewundern* in the previous example is translated as a modal construction: *pouvons admirer*. Since the French reference keeps the finite verbal construction, this checkpoint will also fail.

- **Compounds**: German compounds are a known challenge for SMT systems, because SMT systems do not possess a decomposition module. And when they finally get to be adequately translated, they fail to match the reference translation. For example, the compound noun *Montblanc-Expedition* is translated as *Montblanc expédition*. Since the reference translation was *expédition au Mont Blanc*, only a single n-gram matches, so the score for this checkpoint is very low (1/10).

- **Apostrophe words**: Another case which scores poorly in n-gram-based comparisons are word contractions, which are common in French, among others. This problem occurs mostly in conjunction with other MT errors, such as word choice. Suppose the evaluation tool has to compare the following instances of a pronoun-verb construction: the reference *j' aimerais bien* and the translation hypothesis *je voudrais*. The recall for this instance will be 0, since the system can not appreciate that the two pronouns (*je* and *j'*) are both variations of the first person singular in French. Moreover, the predicates also have different word-forms, although they convey the same meaning.

- **Synonyms**: As the previous example has showed, synonymy is not taken into consideration when comparing n-grams. Therefore, although phrases such as *un splendide après-midi* and *un magnifique après-midi* (EN: a wonderful afternoon) would perfectly match, they only get a score of 3/6.

6 Conclusions

In this paper we have described our experiments with the purpose of evaluating MT systems against a parallel treebank. We have demonstrated that we can improve the evaluation reliability by using manually checked alignments extracted from the treebank. In this way we could get an insight of the weaknesses of our MT system, by referring to problematic linguistic structures in the source language. In order to obtain a systematic classification of the problems, we should analyze the same structures in both languages.

We can conclude that this evaluation method does not offer a complete picture of the system's quality, especially because the output reduces to a number, as in the case of evaluation metrics. The advantage is that the score regards specific linguistic categories, rather than the overall performance of the system. In order to identify the source of the reported errors, a further manual analysis is needed.

The experiments have confirmed the advantage of using in-domain data for training SMT systems. Our system trained on a relatively small amount of in-domain training data (compared to the size of other corpora) outperforms systems not adapted to the domain. The better scores obtained by our SMT system in this evaluation scenario correlate with the BLEU scores reported in [12]. This finding proves the legitimacy of this evaluation approach, which is worthwhile to be extended, in order to obtain a more fine-grained analysis of the MT output.

References

[1] Anne Abeillé, Lionel Clément, and François Toussenel. Building a treebank for French. In *Treebanks : Building and Using Parsed Corpora*, pages 165–188. Springer, 2003.

[2] Wauter Bosma, Piek Vossen, German Rigau, Aitor Soroa, Maurizio Tesconi, Andrea Marchetti, Monica Monachini, and Carlo Aliprandi. KAF: a generic semantic annotation format. In *Proceedings of the 5th International Conference on Generative Approaches to the Lexicon*, 2009.

[3] Elisabet Comelles, Jordi Atserias, Victoria Arranz, and Irene Castellón. VERTa: Linguistic features in MT evaluation. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, 2012. European Language Resources Association (ELRA).

[4] Qin Gao and Stephan Vogel. Parallel implementations of word alignment tool. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, SETQA-NLP '08, pages 49–57, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[5] Anne Göhring and Martin Volk. The Text+Berg corpus: An alpine French-German parallel resource. In *TALN 2011*, July 2011.

[6] Esther König and Wolfgang Lezius. The TIGER language - a description language for syntax graphs - Part 1: User's guidelines, 2002.

[7] Ding Liu and Daniel Gildea. Syntactic features for evaluation of machine translation. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, 2005.

[8] Chi-kiu Lo and Dekai Wu. MEANT: An inexpensive, high-accuracy, semi-automatic metric for evaluating translation utility based on semantic roles. In *Proceedings of ACL HLT 2011*, pages 220–229, 2011.

[9] Sudip Kumar Naskar, Antonio Toral, Federico Gaspari, and Andy Way. A framework for diagnostic evaluation of MT based on linguistic checkpoints. In *Proceedings of the 13th Machine Translation Summit*, pages 529–536, Xiamen, China, September 2011.

[10] Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, March 2003.

[11] Maja Popović and Hermann Ney. Word error rates: decomposition over POS classes and applications for error analysis. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 48–55, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics.

[12] Rico Sennrich. Combining multi-engine machine translation and online learning through dynamic phrase tables. In *EAMT-2011: the 15th Annual Conference of the European Association for Machine Translation*, May 2011.

[13] Rico Sennrich and Martin Volk. MT-based sentence alignment for OCR-generated parallel texts. In *The Ninth Conference of the Association for Machine Translation in the Americas (AMTA 2010)*, 2010.

[14] Martin Volk, Torsten Marek, and Yvonne Samuelsson. Building and querying parallel treebanks. *Translation: Computation, Corpora, Cognition (Special Issue on Parallel Corpora: Annotation, Exploitation and Evaluation)*, 1(1):7–28, 2011.

[15] Ming Zhou, Bo Wang, Shujie Liu, Mu Li, Dongdong Zhang, and Tiejun Zhao. Diagnostic evaluation of machine translation systems using automatically constructed linguistic check-points. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 1121–1128, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

An integrated web-based treebank annotation system

Victoria Rosén,[*§] Paul Meurer,[§] Gyri Smørdal Losnegaard,[*]
Gunn Inger Lyse,[*] Koenraad De Smedt,[*]
Martha Thunes[*] and Helge Dyvik[*§]

University of Bergen[*] and Uni Research[§]
Bergen, Norway

E-mail: victoria@uib.no,paul.meurer@uni.no

Abstract

In recent years a development towards easier access to treebanks has been discernible. Fully online environments for the creation, annotation and exploration of treebanks have however been very scarce so far. In this paper we describe some user needs from the annotator perspective and report on our experience with the development and practical use of a web-based annotation system.

1 Introduction

In the context of earlier semi-automated annotation projects, various annotation tools and interfaces have been created and deployed. User interfaces for efficient parse selection are described in the context of Alpino [23] and Lingo Redwoods [15], inspired by the TreeBanker [7]. Annotation tools for the Paris 7 French treebank were created as extensions of the Emacs editor and provide annotators with interactive editing as well as visualization of trees [1]. Several graphical tree editors and viewers have been developed, such as the type checking dependency tree editor TrEd, used for the Prague Dependency Treebank (PDT) [6, 11], the Annotate tool, used in Negra and Tiger [2, 3], and SALTO, which was developed for SALSA [5]; the latter supports resolution of inter-annotator disagreements. These tools require installation of applications on the client side which may be platform specific.

New developments are oriented towards web-based interaction. The Alpino treebank [23] is browsable on the web with visualizations in SVG (Scalable Vector Graphics), but does not seem to offer further online functionality. The brat annotation system [22] is a web-based tool for text annotation which offers a user-friendly web interface for manual annotation and is very suitable for collaborative editing.

Also, it interacts with external resources and NLP tools. It is well suited for dependency graphs but its visualization does not seem to display constituent trees or (recursive) feature-value graphs, nor does it seem to support parse selection.

In the INESS project, we have probably been the first to develop a fully web-based infrastructure for treebanking. This approach enables annotation as a distributed effort without any installation on the annotators' side and offers immediate online exploration of treebanks from anywhere. In earlier publications, various aspects of this infrastructure have been presented, but specific interface aspects of the annotation process have not been systematically described.

In the remainder of this paper, we will describe some aspects of the annotation interface that we have implemented. Section 2 provides an overview of the INESS project and reviews some of the web-based annotation features that have been described in earlier publications. After that we present new developments: in section 3 we discuss the preprocessing of texts to be parsed in the Norwegian treebank, and in section 4 the integrated issue tracking system is presented.

2 The INESS treebanking infrastructure

INESS, the Norwegian Infrastructure for the Exploration of Syntax and Semantics, is a project cofunded by the Norwegian Research Council and the University of Bergen. It is aimed at providing a virtual eScience laboratory for linguistic research [19]. It provides an environment for both annotation and exploration and runs on its own HPC cluster.

One of the missions of the INESS project is to host treebanks for many different languages (currently 24) and annotated in various formalisms, in a unified, accessible system on the web. Some treebanks are currently only small test suites, while others are quite large, for instance the TIGER treebank, which consists of about 50,000 sentences of German newspaper text. There are dependency treebanks, constituency treebanks, and LFG treebanks; furthermore a number of parallel treebanks are available, annotated at sentence level and experimentally at phrase level [10]. We have implemented middleware for online visualization of various kinds of structures and for powerful search functionality in these treebanks [14].

The second mission of the INESS project is to develop the first large treebank for Norwegian. This treebank is being built by automatic parsing with an LFG grammar, NorGram [8, 9], on the XLE platform [13]. An LFG grammar produces two parallel levels of syntactic analysis; the c(onstituent)-structure is a phrase structure tree, while the f(unctional)-structure is an attribute–value matrix with information about grammatical features and functions [4].

Deep analysis with a large lexicon and a large grammar can often lead to massive ambiguity; a sentence may have hundreds or thousands of possible analyses. Therefore, the LFG Parsebanker was developed in the LOGON and TREPIL projects to enable efficient semiautomatic disambiguation [17, 21]. We have developed discriminants for LFG analyses, as described earlier [20].

In our ongoing annotation work with the LFG Parsebanker, we are basically developing the grammar, the lexicon and a gold standard treebank in tandem [18,

16] and we have earlier reported from this interactive mode of development [12]. In this work it has become clear to us that a tightly integrated development of the grammar, the lexicon and the treebank requires new interactive tools in an integrated annotation environment. Therefore in the current paper we report on the design and implementation of this web-based environment for annotation, focusing on two components which deserve more attention: text preprocessing and issue tracking for treebanking.

3 Text preprocessing

In any treebank of considerable size, identification of words unknown to the lexicon and/or morphology presents a challenge. Although INESS has access to a large lexicon of Norwegian and a sophisticated morphological component, our experience from parsing is that many words in naturally occurring texts are still unknown. Although statistical approaches may attempt to provide the missing lexical information, these are not applicable in our current approach to building a quality treebank, where one incorrectly analyzed word may be enough to cause the parser to fail to produce the correct analysis, even though the syntactic constructions involved in the sentence are within the coverage of the grammar. We have therefore implemented a text preprocessing interface that makes it possible for annotators to quickly identify unknown words and add the necessary properties for treebanking.

An important source of texts for the INESS Norwegian treebank are OCR files from the National Library of Norway. The OCR software makes some errors, such as misinterpreting characters, omitting material, or inserting unwanted material into the text. These problems require efficient preprocessing, which we have implemented in two main stages: initial text cleanup and the treatment of unknown words. These will be discussed in the following sections.

3.1 Initial text cleanup

For the initial text cleanup, the text preprocessing interface presents the text by paragraphs with editing functions. All material that does not belong to the running text itself, such as author, publisher, table of contents, and page numbers, must be removed in text cleanup.

The screenshot in figure 1 shows a text from a book where the last sentence on a page continues on a new page with an illustration in between. The horizontal lines mark the borders between paragraphs. The word *små* 'small' towards the end of the second paragraph and the word *skolepulter* 'classroom desks' at the beginning of the last paragraph are consecutive words in the sentence broken up by the page break. The intervening illustration has caused the insertion of two unnecessary paragraphs with superfluous characters.

The system hyphothesizes that paragraphs that do not end with punctuation are headings, and it marks such paragraphs with background shading. The shading makes it easy for the annotator to check these paragraphs to see if they are indeed headings. In this case, incorrectly split text is the cause of the incorrect marking as a

Figure 1: Screenshot showing OCR-read text in the preprocessing interface

heading. Another common cause of sentences being incorrectly marked as headings is missing end punctuation, a not uncommon OCR error with some fonts.

For each paragraph, a number of buttons provide the most common correction choices: deleting one or more paragraphs, linking paragraphs, editing within a paragraph, adding missing punctuation, marking a paragraph as a title or marking a presumed title as a paragraph. The [Show] button causes the scanned image of the text to appear on the interface; this is useful when it is necessary to check the original.

Unknown words are marked in boldface and shaded with a yellow background in the text, making them easy for the annotator to spot. In figure 1 this is the case with the string *plasert*, an archaic spelling of *plassert* 'placed'.

All unknown words are not only highlighted in the text, but are also presented in an alphabetical list, as shown at the top of figure 2. There is also a list of extracted uncertain (suspicious) words, e.g. the numeral *10*, which is sometimes produced by the OCR software instead of the intended word *lo* 'laughed'. The lists of unknown and uncertain words will shrink when the initial text cleanup is done, so that they only display words that need to be further categorized.

3.2 Categorizing unknown words

Foreign words, named entities, productive compounds, multiword expressions, neologisms, misspellings and variant spellings are examples of the types of words that may not be recognized automatically in the initial preprocessing phase and that need to be categorized. The interface for this processing step is shown in figure 2, where the selection of the string *Bergom* has opened the *Store as* window, in which the annotator can make appropriate choices to correct the word or add it to the lexicon. Each occurrence of a chosen word in the text is shown with its immediate context in the *Context(s)* window. In the case of the string *Bergom*, the contexts make it

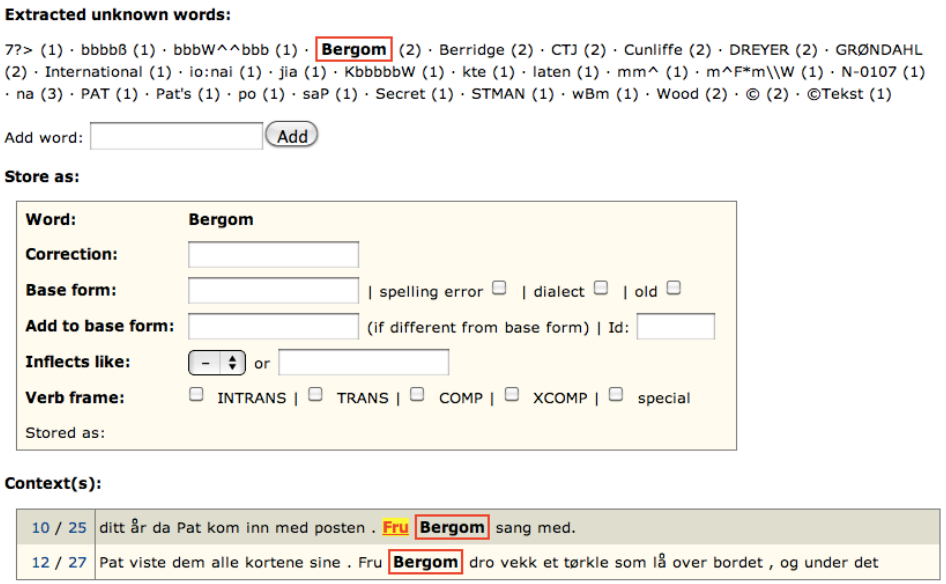


Figure 2: Screenshot of the interface for working with unrecognized words

clear that this is a family name, since in both contexts it is preceded by the word *Fru* 'Mrs.'. For each context, there is a hyperlink to the position of the occurrence in the running text so that the string may be viewed in an even broader context. An additional hyperlink will display the page in the photographic scan of the original, which sometimes reveals that characters are missing in the OCR version.

Typos and OCR errors can be systematically corrected in this interface, even though they can also be edited directly in the text interface in figure 1. Variant spellings that could be common, such as the earlier mentioned *plasert*, may be added to the morphology since we would like to be able to parse the sentence even though the word has a nonstandard orthography. Such spellings also get a special tag in the morphology so that they are marked as deviant; we would not want to produce these forms if the grammar were to be used for generation.

The context will sometimes suggest that the word is part of a multiword expression. The annotator can then select adjacent words to be added as a fixed multiword entity. Not all multiwords may be treated in this way, of course; some require special treatment in both the lexicon and the grammar. Identifying multiword entities is important with respect to parsing, since those unrecognized by the morphology will be analyzed compositionally by the parser unless they have already been entered into the LFG lexicon.

Not all OCR, typographical and spelling errors are captured by morphological preprocessing. This may be for instance because the string is an existing word. For example, when the word form *term* 'term' occurs in the context *term lyset* 'term the light', it is clear to a native speaker that the original text must have read *tenn lyset* 'light the light'. This may be confirmed by consulting the image of the original text. Such errors may be spotted both during text cleanup and during further word

processing. The *Add word* function allows the annotator to type in words that the system has missed and add them to the list of unknown words. The same functionality for adding words to the morphological component has also been implemented in the disambiguation interface, so that when an error like this is discovered *after* parsing, it may be added to the morphology in the same way and treated correctly in a reparse of the sentence.

3.3 Assigning lexical properties to unknown words

If an unrecognized word must be added to the lexicon, the annotator has to decide whether it is a new word, to be added to the morphology as a new paradigm, or a variant form of an existing paradigm. Entering new noninflecting words is a straightforward process involving simply choosing the correct word category, such as interjection, place name, masculine first name, feminine first name, etc.

Words belonging to the open, productive word classes (nouns, verbs, adjectives and adverbs) usually have inflectional paradigms which must be defined. The dictionary entry form of the word is entered in the interface as the *Base form*, and a similar word with the same word class and inflection pattern is specified, either by selecting it from a drop-down list of suggestions, or by entering it into a text box. The system proposes as candidates a number of word forms ending with the same characters at the end of the word. When one of these candidates is chosen, the system automatically generates the paradigm for the word being entered. In figure 3, the new compound *erkemikkelen* 'arch fool' is being added as a paradigm inflecting like the existing compound *dåsemikkel* 'nitwit', and the new paradigm is shown to the right. If the annotator decides that this is the correct inflection and chooses this paradigm, all inflected forms of the unrecognized word are automatically added to the morphological analyzer.

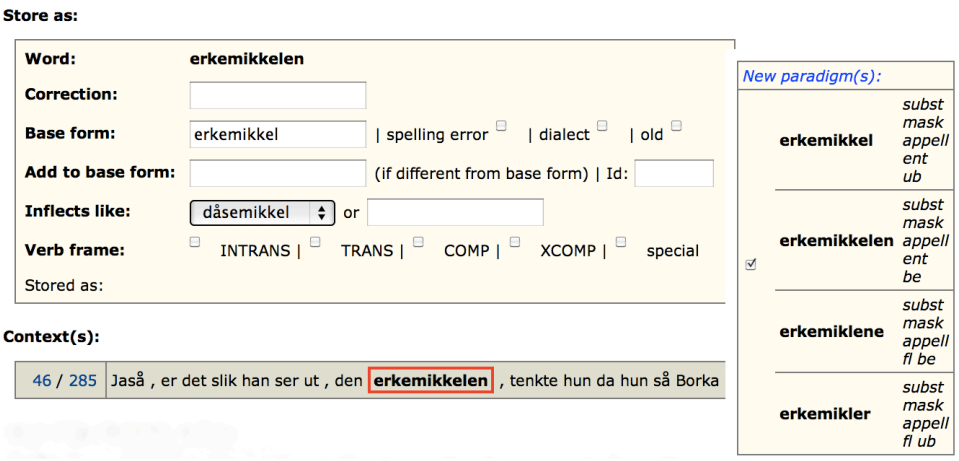


Figure 3: Adding a paradigm

If the new word is a verb, it is not sufficient to add an inflectional paradigm. Verbs must also be assigned subcategorization frames necessary for parsing. Sim-

ple subcategorization frames may be added by ticking one or more of the boxes INTRANS (intransitive), TRANS (transitive), COMP (sentential complement) or XCOMP (infinitival complement). In some cases the subcategorization frame may be more complicated, as in the case of particle verbs. These must be indicated by ticking the box *special*, and they will then be further processed manually.

Norwegian is a language that allows many alternative forms within the officially recognized orthography, both in stems and in endings. The extensive variation in the official language is already represented in the existing Norwegian lexicon used together with our grammar. Additional, marked variants must be classified into three main categories called *misspelling*, *dialect* and *old*. Although these variants are not part of the official language, we want to be able to analyze them, and therefore enter them as variant forms in the appropriate inflectional paradigms. Sometimes words are unintentionally or intentionally misspelled by the author, for instance to represent a child's spelling errors. These are classified as *misspelling*. In addition to spelling errors, there are other sources of unconventional orthography. Especially in novels, there are many cases where the author has wanted to imitate special language forms, either to portray dialectal variation or a particular accent. Such cases are classified as *dialect*. Finally, some unrecognized spellings may be archaic forms, which are classified as *old*.

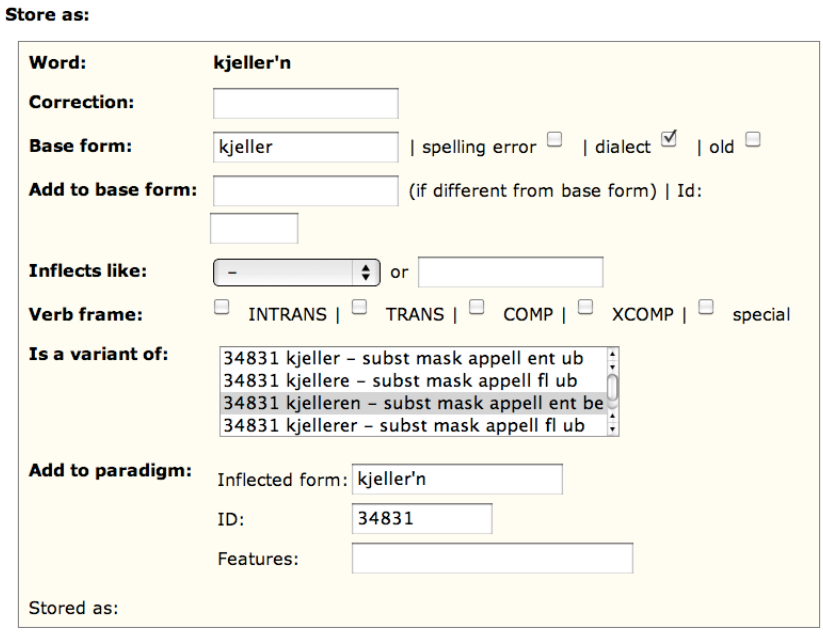


Figure 4: Adding word form variants

There are two different ways of entering variants into the morphology, dependent on whether the deviation from the standard written form pertains to the inflectional ending or the stem. When the variation may be regarded as only a de-

viant ending, the correct base form (lemma) is entered, and a shortcut brings up the paradigm(s) associated with the base form (the field *Is a variant of* in figure 4). The paradigm is then presented as a list of all existing word forms with their morphological features, from which the annotator selects one or more paradigm rows with the appropriate features. If there is no such set of features, as will be the case when a word is used with deviating gender, the features must be typed in manually in the *Features* field. In the example in figure 4, the word *kjelleren* 'the basement' has been spelled with an apostrophe rather than the vowel *e*, representing a common pronunciation where the schwa in the final syllable has been dropped.

When the variation concerns the spelling of the stem, an entire paradigm is added to the morphology. In figure 5, the misspelling *kolapsa* is being added to the paradigm for *kollapse* '(to) collapse'. This error is made systematically throughout the text and is probably intentional (not a typo), and it is also likely to be a common mistake. The word is added to the morphology by entering the base form of the variant, *kolapse*, in the *Base form* field, and then typing in the base form of the standard (*Add to base form*). All possible paradigms appear in the box to the right (in this particular case only one) and the appropriate paradigm is chosen.

All extracted words are stored in a database together with their assigned lexical properties and the context they were extracted from. Here, they can be reviewed and reclassified/edited if necessary. Before the texts those words are extracted from are added to the treebank and parsed, the extracted words and their paradigms have to be added to the morphology used in the LFG grammar. Since this add-on morphology is not technically merged with the main morphology, but compiled as a separate transducer, the main morphological transducers do not have to be recompiled, and updating of the add-on morphology is done in a matter of seconds.

Figure 5: Adding stem variants

4 Integrated interface for annotation and issue tracking

An efficient working environment is essential in a treebanking annotation system. The annotator must be able to check previous analysis choices and communicate with colleagues on the desired annotation. Although an external project management system such as Redmine can be used for creating and tracking issues, the disadvantage is that the link between the issue and the annotated sentence is at best indirect and involves switching systems. A better approach, which we have pursued, and which owes many ideas to the Redmine system, is to have one single user interface for annotation and issue tracking. This approach also offers opportunities for tailoring the tracking system to the treebanking task.

We have implemented this in such a way that issues may be handled directly on the same webpage that is used for sentence annotation. When the annotator finds that the correct analysis is not available, or is unsure about which analysis to choose, an issue is entered into a comment window. Each issue is assigned a *category*, a *subject* and a *status*. Examples of *categories* are major types of coverage problems, such as *grammar update*, *lexicon update*, *MWE* (*multiword expression*), and *annotator question*, used when the annotator is unsure of the desired analysis. *Subjects* are subcategories. For instance, *lexicon update* has the subcategories *lexical frames*, *new word*, *new sense*, *named entity* and *morphology*. Status can be *new*, *open*, *pending* or *closed*.

The system automatically sends e-mail notifications whenever a new posting in the comment field is made. The e-mail includes the most recent posting and a hyperlink that leads directly to the sentence annotation page, where all postings in the comment field may be viewed *together* with the analysis of the sentence. In addition to sentence and parse related comments, the system can also handle comments and issues that are related to a specific treebank, or to the platform in general. Such issues may include bug reports, feature requests, or any other type of general discussion.

It is important for consistency that similar constructions are annotated in the same way. When an annotator wants to check how similar cases have been annotated earlier, various types of searches may be made using the INESS-Search facility [14]. The analyses in the treebank may be searched by formulating a search expression in the query window or by choosing a previously saved search from a menu. Such searches may be made for instance for certain grammatical constructions, such as headless relatives, or for features of analyses, such as temporal nouns.

Searches may also be made in previous comments. Figure 6 shows the interface for searching among comments. In this interface the annotator may specify words or strings in both the comments and in the treebank texts, and the issue category, subject and status may also be specified if desired.

The system for identifying different types of issues in annotation is not only important as an aid to the annotator in deciding on the correct annotation, it is also useful for studying the performance of the system as a whole. The reason(s) why a sentence does not have the desired analysis may be many and varied, and this issue tracking system makes it possible to study the distribution of coverage problems.

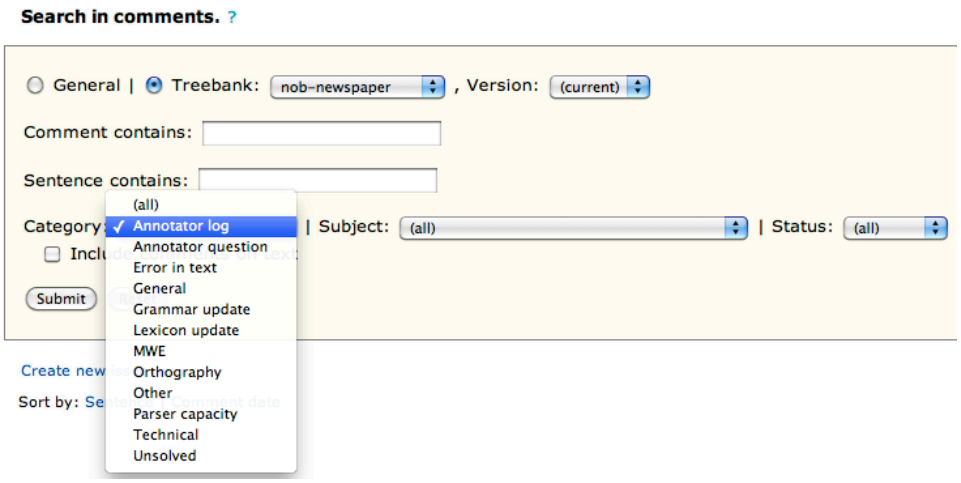


Figure 6: Comment search interface

5 Conclusion

Based on our experience, an integrated web interface promotes interaction between annotators, the grammar writer, the web application developer and the project leader. It provides a seamless environment both for building the treebank and for inspecting it. It allows distributed annotation in real time without the need to install client systems other than a web browser. The system presented here represents work in progress; it will be further developed as we gain more experience with annotation.

References

[1] Anne Abeillé, Lionel Clément, and François Toussenel. Building a treebank for French. In Anne Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*, volume 20 of *Text, speech and language technology*. Kluwer Academic Publishers, Dordrecht, 2003.

[2] Thorsten Brants and Oliver Plaehn. Interactive corpus annotation. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC)*, 2000.

[3] Thorsten Brants, Wojciech Skut, and Hans Uszkoreit. Syntactic annotation of a German newspaper corpus. In Anne Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*, volume 20 of *Text, speech and language technology*, pages 73–87. Kluwer Academic Publishers, Dordrecht, 2003.

[4] Joan Bresnan. *Lexical-Functional Syntax*. Blackwell, Malden, MA, 2001.

[5] Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, and Sebastian Pado. SALTO – a versatile multi-level annotation tool. In *Proceedings*

of the 5th International Conference on Language Resources and Evaluation (LREC), pages 517–520. European Language Resources Association (ELRA), 2006.

[6] Tomas By. Graphical, type-checking dependency tree editor. In *Proceedings of the International Multiconference on Computer Science and Information Technology*, pages 179–184, 2009.

[7] David Carter. The TreeBanker: A tool for supervised training of parsed corpora. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pages 598–603, Providence, Rhode Island, 1997.

[8] Helge Dyvik. The universality of f-structure. Discovery or stipulation? The case of modals. In Tracy Holloway King and Miriam Butt, editors, *Proceedings of the LFG99 Conference*, Stanford, 1999. CSLI Publications.

[9] Helge Dyvik. Nødvendige noder i norsk. Grunntrekk i en leksikalsk-funksjonell beskrivelse av norsk syntaks [Necessary nodes in Norwegian. Basic features of a lexical-functional description of Norwegian syntax]. In Øivin Andersen, Kjersti Fløttum, and Torodd Kinn, editors, *Menneske, språk og felleskap*. Novus forlag, 2000.

[10] Helge Dyvik, Paul Meurer, Victoria Rosén, and Koenraad De Smedt. Linguistically motivated parallel parsebanks. In Marco Passarotti, Adam Przepiórkowski, Sabine Raynaud, and Frank Van Eynde, editors, *Proceedings of the Eighth International Workshop on Treebanks and Linguistic Theories*, pages 71–82, Milan, Italy, 2009. EDUCatt.

[11] Jan Hajič, Barbora Vidová-Hladká, and Petr Pajas. The Prague Dependency Treebank: Annotation structure and support. In *Proceedings of the IRCS Workshop on Linguistic Databases*, pages 105–114, Philadelphia, 2001.

[12] Gyri Smørdal Losnegaard, Gunn Inger Lyse, Martha Thunes, Victoria Rosén, Koenraad De Smedt, Helge Dyvik, and Paul Meurer. What we have learned from Sofie: Extending lexical and grammatical coverage in an LFG parsebank. In Jan Hajič, Koenraad De Smedt, Marko Tadić, and António Branco, editors, *META-RESEARCH Workshop on Advanced Treebanking at LREC2012*, pages 69–76, Istanbul, Turkey, May 2012.

[13] John Maxwell and Ronald M. Kaplan. The interface between phrasal and functional constraints. *Computational Linguistics*, 19(4):571–589, 1993.

[14] Paul Meurer. INESS-Search: A search system for LFG (and other) treebanks. In *Proceedings of the LFG '12 Conference*, 2012 (forthcoming).

[15] Stephan Oepen, Dan Flickinger, Kristina Toutanova, and Christopher D. Manning. LinGO Redwoods: A rich and dynamic treebank for HPSG. *Research on Language and Computation*, 2(4):575–596, December 2004.

[16] Victoria Rosén and Koenraad De Smedt. Theoretically motivated treebank coverage. In *Proceedings of the 16th Nordic Conference of Computational Linguistics (NoDaLiDa-2007)*, pages 152–159. Tartu University Library, Tartu, 2007.

[17] Victoria Rosén, Koenraad De Smedt, Helge Dyvik, and Paul Meurer. TREPIL: Developing methods and tools for multilevel treebank construction. In Montserrat Civit, Sandra Kübler, and Ma. Antònia Martí, editors, *Proceedings of the Fourth Workshop on Treebanks and Linguistic Theories (TLT 2005)*, pages 161–172, 2005.

[18] Victoria Rosén, Koenraad De Smedt, and Paul Meurer. Towards a toolkit linking treebanking to grammar development. In *Proceedings of the Fifth Workshop on Treebanks and Linguistic Theories*, pages 55–66, 2006.

[19] Victoria Rosén, Koenraad De Smedt, Paul Meurer, and Helge Dyvik. An open infrastructure for advanced treebanking. In Jan Hajič, Koenraad De Smedt, Marko Tadić, and António Branco, editors, *META-RESEARCH Workshop on Advanced Treebanking at LREC2012*, pages 22–29, Istanbul, Turkey, May 2012.

[20] Victoria Rosén, Paul Meurer, and Koenraad De Smedt. Designing and implementing discriminants for LFG grammars. In Tracy Holloway King and Miriam Butt, editors, *The Proceedings of the LFG '07 Conference*, pages 397–417. CSLI Publications, Stanford, 2007.

[21] Victoria Rosén, Paul Meurer, and Koenraad De Smedt. LFG Parsebanker: A toolkit for building and searching a treebank as a parsed corpus. In Frank Van Eynde, Anette Frank, Gertjan van Noord, and Koenraad De Smedt, editors, *Proceedings of the Seventh International Workshop on Treebanks and Linguistic Theories (TLT7)*, pages 127–133, Utrecht, 2009. LOT.

[22] Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. brat: a web-based tool for NLP-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107, Avignon, France, April 2012. Association for Computational Linguistics.

[23] Leonoor van der Beek, Gosse Bouma, Robert Malouf, and Gertjan van Noord. The Alpino dependency treebank. In *Computational Linguistics in the Netherlands (CLIN) 2001*, Twente University, 2002.

Translational divergences and their alignment in a parallel multilingual treebank*

Manuela Sanguinetti, Cristina Bosco

Università di Torino
Department of Computer Science
E-mail: {msanguin;bosco}@di.unito.it

Abstract

The usefulness of parallel corpora in translation studies and machine translation is strictly related to the availability of aligned data. In this paper we discuss the issues related to the design of a tool for the alignment of data from a parallel treebank, which takes into account morphological, syntactic and semantic knowledge as annotated in this kind of resource. A preliminary analysis is presented which is based on a case study, a parallel treebank for Italian, English and French, i.e. ParTUT. The paper will focus, in particular, on the study of translational divergences and their implications for the development of an alignment tool of parallel parse trees that, benefitting from the linguistic information provided in ParTUT, could properly deal with such divergences.

1 Introduction

Parallel corpora are currently considered as crucial resources for a variety of NLP tasks (most notably machine translation), and for research in the field of translation studies and contrastive linguistics. In order to be of any help for such purposes, parallel corpora have to be correctly aligned, that is to say that translational correspondences between the pairs of languages involved should be properly detected and exploited. Several approaches have been used in order to automatically align these multilingual resources, based both on deterministic rules or specific heuristics (see, e.g. [11]), and statistical techniques (e.g. [9], and [16]). The latter in particular have been highly successful in recent years. The reasons for such success are manyfold: among them we can find their capability to process even less-studied or resource-poor languages, or the large amount of time required to create robust and accurate rule-based systems. It is our belief, however, that linguistic insights can

*This work has been partially funded by the PARLI Project (Portale per l'Accesso alle Risorse Linguistiche per l'Italiano - MIUR - PRIN 2008).

be of great help and that the presence of a set of rules for the detection of translational correspondences can, if not replace, significantly complement the work done by statistical systems. Linguistic knowledge can be of help in identifying not only the exact matches, but also (we would rather say in particular) all those cases in which there are partial or fuzzy correspondences due, for example, to the individual translator choices or to differences - which often occur in a systematic way - between language pairs.

In this paper we focus our attention especially on these differences (which we will designate with the term "shift"), on their classification, and finally on a proposal for their automatic processing. We start, in Section 2, from the discussion of some works in the field of Translation Studies where reference is made to the notion of translation shift, and we then present an existing resource, i.e. ParTUT, which has been used as the basis for our analysis. The remainder of the paper is structured as follows: in Section 3 we provide a brief description of the corpus content and of the annotation format of the treebank, while Section 4 is devoted to the cases of translation shifts encountered so far in the corpus and to the identification of the main points our research should be addressed to.

2 Related Work

Due to the peculiarities of each language system, translation process may be quite complex and correspondences could be drawn up on various linguistic levels: lexical, structural or semantic; it therefore entails the need to implement strategies (or "shifts") so that the meaning is properly transposed and preserved in such process.

John Catford [1] was the first one to use the term *shift*, which he defined as the departure from formal correspondence in the process of going from the source language to the target language. His definition relied upon the notions of *formal correspondence* and *textual equivalence*. A formal correspondence is a relationship between two linguistic items that play the same role in their respective systems, while textual equivalence is defined, in a broader sense, as any kind of translational equivalence in a pair of texts or sentences. A shift then occurs in translation whenever there is not a formal correspondence relationship between two elements in source and target texts. In his work, Catford discussed two types of shift: level and category shift. The former deals with shifts between different linguistic levels (usually grammar and lexis), while category shift is further subdivided in: class shift (e.g. in the word class used), unit or rank shift (e.g. when a single word is translated by means of a phrase), structural shift (e.g. when word order is modified) – this is considered the most frequent among the category shifts – and intra-system shift (i.e. when, despite the presence of a formal correspondence between source and target elements, a non-corresponding form is chosen while translating).

An important contribution which largely relied upon the linguistic theories described above is that of Cyrus [3], who explicitly annotated translation shifts in a parallel treebank. Contrarily to most of the works on parallel treebanks pro-

posed so far, Cyrus' work did not aim at creating a machine translation system, but at building a resource for English and German (FuSe) in which translation shifts were explicitly annotated on the basis of the predicate-argument structure. The annotation system was based on the distinction between two main classes, i.e. grammatical and semantic shifts. In the first one, all those cases of passivisation–depassivisation, pronominalisation–depronominalisation, as well as category and number changes were included. Semantic shifts comprised the cases when meaning is somewhat involved and were classified as follows: semantic modification, explicitation– generalisation, addition–deletion and mutation.

These works have been an important theoretical reference for our research. The analysis of translational correspondences and divergences made on our small set of sentences, besides taking into account the observed cases, as well as some peculiarities of the corpus (such as the annotation format), has been largely inspired by such works in its theoretical formalization and systematization.

3 Corpus description

3.1 Data

ParTUT[1] currently comprises 42,347 tokens distibuted as shown in Table 1. The corpus consists of an average amount of 465 sentences per language. They were retrieved from the JRC-Acquis multilingual parallel corpus[2] [15] and the entire text (about 100 sentences) of the Universal Declaration of Human Rights[3]. More recently this preliminary set has been enlarged with an additional corpus extracted from the open licence "Creative Commons"[4] composed by around 100 sentences, and from publicly available pages from Facebook website.

The corpus is aligned on the sentence level with the Microsoft Bilingual Sentence Aligner ([12]) and the LF Aligner[5], an automatic tool based on Gale and Church algorithm which enables the storage of sentence pairs as translation units in TMX files and the review of the output in formatted xls spreadsheets.

3.2 The annotation format

The parallel treebank comprises a collection of sentences represented in the form of dependency structures. The dependency relations are described in compliance with the same criteria adopted for the creation of the Italian monolingual treebank TUT (Turin University Treebank)[6].

[1] http://www.di.unito.it/tutreeb/partut.html
[2] http://langtech.jrc.it/JRC-Acquis.html, http://optima.jrc.it/Acquis/
[3] http://www.ohchr.org/EN/UDHR/Pages/SearchByLang.aspx
[4] http://creativecommons.org/licenses/by-nc-sa/2.0
[5] http://sourceforge.net/projects/aligner/
[6] http://www.di.unito.it/tutreeb/

Corpus	sentences	tokens
JRCAcquis–It	181	6,304
JRCAcquis–En	179	4,705
JRCAcquis–Fr	179	8,667
UDHR–It	76	2,387
UDHR–En	77	2,293
UDHR–Fr	77	2,537
CC–It	96	3,141
CC–En	88	2,507
CC–Fr	102	3,624
FB–It	115	1,947
FB–En	114	1,723
FB–Fr	112	2,512
total	1,377	42,347

Table 1: Corpus overview.

As far as the native annotation schema is concerned, a typical TUT tree shows a pure dependency format centered upon the notion of argument structure and is based on the principles of the *Word Grammar* theoretical framework [7]. This is mirrored, for instance, in the annotation of Determiners and Prepositions which are represented in TUT trees as complementizers of Nouns or Verbs. By contrast, the native TUT scheme exploits also some representational tools, i.e. null elements (which are non–standard in dependency-based annotations), in order to deal with particular structures, such as pro–drops, long distance dependencies and elliptical structures.

As for the dependency relations that label the tree edges, TUT exploits a rich set of grammatical items designed to represent a variety of linguistic information according to three different perspectives, i.e. morphology, functional syntax and semantics. The main idea is that a single layer, the one describing the relations between words, can represent linguistic knowledge that is proximate to semantics and underlies syntax and morphology, i.e. the predicate-argument structure of events and states. To this end, a distinction is drawn between modifiers and subcategorized arguments on the one hand and between surface and deep realization of any admitted argument; these annotation criteria have proven particularly useful while detecting cases of nominalisations and passivisation, which are common cases of divergence in translation.

In a cross-linguistic perspective, the choice to use a single and coherent representation format, and the TUT format in particular, proved to be suitable in the development of a parallel resource in that the rich morphological tag set allowed an adequate representation for both morphologically rich languages (Italian and French) and simpler ones (English), and the richness and flexibility of relations

allowed an appropriate coverage of new linguistic phenomena encountered so far.

Such format proved also to be useful in comparative analyses, by means of which some typical phenomena of the three languages involved could be easily queried and quantified. Among these we can find the higher frequency of nominal modifiers (4.5%) in English texts (expressed by the NOUN-RMOD label), with respect to Italian (0.9%) and French (0.6%), or, on one hand, the presence (marked, as explained above, by null elements) of pro-drops in Italian sentences (56 occurrences against one single entry in English and the absence in French) and, on the other, the use of expletive subjects in English and French (respectively 19 and 21 occurences). Furthermore, the annotation of Determiners as complementizers of Nouns (as also pointed out above) and the lower frequency of determiners in English (11.6%, compared to 13.4% in French and 16.1% in Italian) led to a different representation of English trees with respect to Ialian.

These preliminary considerations, together with the absence of an appropriate tool that allows the alignment of sentences represented according to the TUT format, have led to the decision of working on the development of a new system. As, in fact, the choice of the aligment tool is strongly related to the final task, we cannot abstract away from the input format, especially if it provides linguistically relevant information which can be useful for any further corpus processing and exploitation.

4 Alignment issues

In this section we describe the main steps of the investigation which has lead us to the definition of our approach for the alignment of the trees in ParTUT.

As a first step, we selected a small subset of sentences from the collection and attempted to align them manually, in order to specifically study the various issues that could arise, and in particular translation shifts. We selected for that purpose the first two subcorpora of the treebank, i.e. the JRC-Acquis and the UDHR, which were already aligned on the sentence level. In order to avoid further drawbacks deriving from multiple or null correspondences – which were caused in some cases by the different segmentation criteria adopted by the parser and the sentence aligner – we only worked on 1:1 sentence pairs (constituting almost the 90% of the overall amounts of sentence pairs). In particular, the experimental corpus was composed by 50 sentences per language divided into three pairs (i.e. Italian - English, English - French, French - Italian). While comparing the tree pairs, we observed the various cases of divergences, or shifts, and attempted to classify them.

Similarly to what proposed in [3], we identified two main classes of shifts, each one involving respectively morpho-syntactic and structural level on one hand, and semantic level on the other. In the first class we include:

Category shift[7]: when different parts of speech are used in source and target text.

[7] Although Catford used this term for describing a main class of shifts that included other sub-

EN: ***Improving*** *the efficiency* [...]
FR: ***L'amélioration de*** *l'efficacité* [...]
(*The improvement of the efficiency*)[8]

Structural shift: this subclass comprises all those cases when syntactic level is directly involved and affected from translator's choices or word order constraints. We then include, for example, the cases of discontinuous correspondences:

EN: *Nor shall* ***a heavier penalty*** *be imposed* ***than*** *the one that was applicable* [...]
IT: *Non potrà del pari essere inflitta* ***alcuna pena superiore a*** *quella applicabile* [...]
(Cannot be likewise imposed any penalty heavier than the one applicable)

passivisation–depassivisation[9]:

EN:*This Directive* ***seeks to*** *achieve* [...]
IT:*La presente Direttiva* ***è intesa a*** *conseguire* [...]
(The present Directive is sought to achieve)

different syntactic realizations (e.g. light verb constructions or paraphrases):

EN: [...] ***to achieve the promotion*** *of respect* [...]
IT: [...] ***promuovere*** *il rispetto* [...]
(to promote the respect)

As already observed in [1], this is the most frequent type of translation shift and, we would rather say, the most evident, when comparing a tree pair.
The second main class often involves some of the structural shifts as well; despite this, we preferred to classify them separately. They may include:

addition–deletion:

EN: [...] *the respect for* ***and observance of*** *human rights and fundamental freedoms* [...]
FR: [...] *le respect* ***universel et effectif*** *des droits de l'homme et des libertées fon-*

classes, similarly to [3], with this expression we prefer to indicate only morpho-syntactic categories.

[8]The glosses for non-English examples are intended as literal and not necessarily corresponding to the correct English expression.

[9]Since in ParTUT translation direction is unknown, we consider the two transformation strategies as counterparts one of each other and put them in the same subclass. We applied the same principle even for the cases of additiondeletion, cited below.

damentales [...][10]
(the universal and effective respect of human rights and of fundamental freedoms)

mutation: whenever there is a textual equivalence (according to Catford's terminology), but the correspondence is characterised by a high degree of fuzziness.

EN: *the right to* ***recognition as a person before the law***
FR: *le droit à la* ***reconnaissance de sa personalité juridique***
(*the right to the recognition of his legal personality*)

Although to a different extent, both classes of shift (i.e. syntactic and semantic) were equally relevant, being the most prominent the structural shifts (with an overall frequency rate of 47.6%), the addition-deletion semantic sub-class (21%) and the category shifts (16.8%)[11]. The description of such shifts in quantitative terms, besides their classification, may provide an insight on the extent to which it is necessary to take into account such differences while designing an automatic alignment system, as well as on the impact that they may have on its overall performance.

4.1 Alignment proposals

Our proposal is based on some fundamental criteria which depend on the observation of the translation shifts. First of all, since in the shifts both morphology and syntax can be involved, we decided that we have to take into account at the same time a morphological and syntactic perspective, as allowed by a format encoding like TUT, with linguistic knowledge for the lexical morphology, dependency structure and argument structure. Observing the several guidelines (see [9] and [12]) and related works (e.g. [7], or [16]) consulted before we started the manual annotation of translational correspondences, we found only one study [15] which dealt with alignment of tree pairs in terms of phrases, rather than single words, and no one where the alignment was based on dependency relations. Useful information on the alignment of argument structure in parallel treebank were instead found in [3] and [6], but in all these guidelines there were significant discrepancies and divergences in the choice of what should actually be put into correspondence.

Our proposal includes two distinct steps, respectively referring to the lexical level and to syntactic dependencies. As for the alignment on the word level, we first used the WordAligner, a web-based interface which allows for manual editing and browsing of alignments and represents each pair of sentences as a grid of squares. For the syntactic level, we worked on an alignment procedure that could then be formalized and implemented benefitting from the syntactic information

[10] In this example, in particular, we observe both additions and deletions while comparing the English sentence to the French version.

[11] And among them, respectively, the shift between name and adjective, (36.3%), and between noun and verb (21.2%).

provided by the annotation. This procedure, as currently conceived, consists of two distinct steps.

Step 1: in a first stage, lexical correspondences are identified and stored by means of a probabilistic dictionary. We obtained such resource by running the same tool used for the sentence alignment, i.e. the Microsoft Bilingual Sentence Aligner (see Section 3.1), which contains an implementation of the IBM Model One. Since data gathered in ParTUT could not be sufficient to obtain reliable results, we ran the tool bidirectionally with texts retrieved from different parallel resource, i.e. Europarl and the Web Inventory of Transcribed and Translated Talks (WIT3)[12][2] (see Table 2 for details on the Italian-English pair).

Source	sentences	tokens
Europarl En	56,057	1,437,469
Europarl Ita	56,057	1,515,431
WIT3 En	9,266	176,345
WIT3 Ita	9,315	175,496

Table 2: Some preliminary results on the creation of a lexical resource for Italian and English with the IBM Model 1 implementation of the Microsoft Bilingual Sentence Aligner.

Step 2: starting from the lexical pairs obtained in the first step, correspondences between neighbouring nodes are verified by means of the information provided by the annotation (i.e. morpho-syntactic category and dependency relations). Such information is useful to predict alignment links between nodes that were not detected as translational equivalents by means of the dictionary, then completing the tentative alignment performed in the previous step.

The procedure, in its general approach (i.e. without considering more specific cases), is described below (see Algorithm 1). For each lexical pair between source and target sentences retrieved in the first step, referred to as *lex_pair(s,t)* in the pseudo-code, the algorithm iteratively searches for head and dependents of the source node *s* in the lexical pair and verifies, at first attempt, whether they belong to other lexical pairs; otherwise, it looks for their linguistic features, first Part of Speech (*PoS*), then syntactic relations (*dep_rel*), and compares them with the corresponding features of head and dependents of *t*.

The choice to use the relational and categorical information proved useful in the identification and resolution of some cases of translation shifts.

As for categorical shifts, for example, a common case is that of nominalization: this is easily detectable by means of the third step of the algorithm, since deverbal nouns are usually annotated as such in TUT exploiting the NOUN-OBJ relation (see Figure 1).

[12] https://wit3.fbk.eu/

for all lex_pair(s,t) **do**
  if head and dependents of s are included in other lex_pair(s,t) **then**
    align nodes
  else if their PoS is the same **then**
    align nodes
  else if their dep_rel are the same **then**
    align nodes
  end if
end for

Algorithm 1: Node alignment procedure after the detection of lexical correspondences.

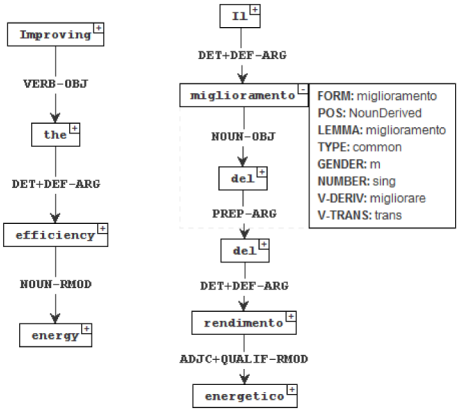


Figure 1: An example of nominalization (in the Italian version) and its annotation.

Even the other most common cases of shifts, i.e. those between adjective and name (whenever they are both modifier of the head node), are easily solved and formalized with the rule displayed above.

Also with regard to structural shifts, the systematic nature of some differences allowed the treatment of those cases with simple rules. For example, the conflation of modal, or a compound verb (especially in English), and the main verb in a single verb. The same applies to cases of change in word order, e.g. because of pre-and post-modification differences, the first being more common in English and the second in French and Italian, as briefly discussed in3.2 (see Figure 2 for an example). This is mainly due to the choice of a representation format that focuses on the dependency relation between a head and its modifier(s), rather than on constituents and their order.

For the same reason, other structural shifts involving word order were equally solvable, although they were less systematic and more subject to stylistic or individual translator's choice, as the following example shows.

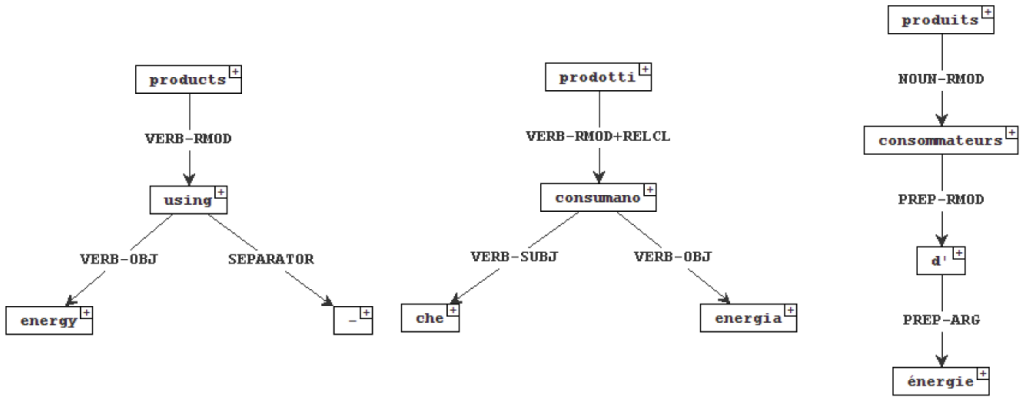


Figure 2: Representation of the expression "*energy-using products*" respectively in English, Italian and French. As observed, the heads in the sub-trees are the same, despite their positions in the three sentences.

EN: *The exchange of information on environmental life cycle performance and on the achievements of design solutions should be facilitated.*
IT: *Dovrebbero essere agevolati uno scambio di informazioni sull'analisi della prestazione ambientale del ciclo di vita e sulle realizzazioni di soluzioni di progettazione.*
(*should be facilitated an exchange of information on the analysis of the environmental life cycle perfomance and on the achievements of design solutions*.)

Even the cases of passivisation–depassivisation, due to the information encoded in the TUT format on the function of verbal arguments, may receive an alignment link: if lexical match is set in the first phase between the two verbs involved, the relational labels of their respective arguments (eg. [VERB-SUBJ] for the subject of the active form, and [VERB-OBJ/VERB-SUBJ] for the surface object of the passive form) were checked, and argument nodes are aligned.

These are all cases that show how and when linguistic knowledge and the linguistic information provided by the processing tools could manage to deal with systematic differences between two languages and with translation divergences.

5 Conclusion and future work

This paper aims to present a study for the design and development of a tool for the alignment of parallel data that exploits the linguistic information (especially on morpho-syntactic and relational level) explicitly annotated in a treebank. The purpose of this study was to investigate the usefulness of such information to overcome the limitations of alignment tools which are not linguistically motivated in

the treatment of typical translational divergences, or shifts. We described in what terms linguistic insights (such as the knowledge of some systematic differences between the language pairs involved), and the choice to use a representation format that focuses on the dependency relations and the predicative structure allows us to deal with some of these shifts by applying simple rules. Other aspects, however, are yet to be fully explored (although partially in progress for the time being), such as the creation of a reference alignment corpus, a systematic evaluation of the overall system and a comparison with the state-of-the-art tools, and the extension of the treebank to other text types in order to obtain a more balanced corpus on one hand, and to verify whether and to what extent the translation shifts classification here proposed, as well as the rules originally conceived for their automatic alignment are still valid and appropriately implemented.

References

[1] Catford, John C. (1965) *A Linguistic Theory of Translation: An Essay on Applied Linguistics*. Oxford: Oxford University Press.

[2] Cettolo, Mauro, Ghirardi, Federico and Federico Marcello (2012) WIT3: A Web Inventory of Transcribed Talks. In *Proceedings of the 16th EAMT Conference*, Trento, Italy.

[3] Cyrus, Lea (2006) Building a Resource for Studying Translation Shifts. In *Proceedings of Language Resources and Evaluation Conference* (LREC'06), Genova, Italy.

[4] Cyrus, Lea (2009) Old concepts, new ideas: approaches to translation shifts. In *MonTI. Monografías de Traducción e Interpretación.*, N. 1 (2009).

[5] Dyvik, Helge, Meurer, Paul, Rosén, Victoria and De Smedt, Koenraad (2009) Linguistically Motivated Parallel Parsebanks. In *Proceedings of the Eighth International Workshop on Treebanks and Linguistic Theories (TLT8)*, Milan, Italy.

[6] Graça, João , Pardal, Joana Paolo, Coheur, Luísa and Caseiro, Diamantino (2008) Building a golden collection of multi-language word alignment. In *Proceedings of Language Resources and Evaluation* (LREC'08), Marrakech, Morocco.

[7] Hudson, Richard (1984) *Word Grammar*. Oxford: Blackwell.

[8] Lambert, Patrik, de Gispert, Adrià, Banchs, Rafael E. and Mariño, José B. (2005) Guidelines for word alignment evaluation and manual alignment. In *Language Resources and Evaluation*, December 2005, Volume 39, Issue 4.

[9] Lavie, Alon, Parlikar, Alok, and Ambati, Vamshi (2008) Syntax-driven Learning of Sub-sentential Translation Equivalents and Translation Rules from Parsed Parallel Corpora. In *Proceedings of the Second Workshop on Syntax and Structure in Statistical Translation* (SSST-2), Columbus, OH.

[10] Melamed, Daniel (1998) *Manual annotation of translational equivalence: The BLINKER project. Technical report #98-07.* Institute for Research in Cognitive Science, University of Pennsylvania, Philadelphia, PA.

[11] Menezes, Arul and Richardson, Stephen D. (2001) A best-first alignment algorithm for automatic extraction of transfer mappings from bilingual corpora. In *Proceedings of the workshop on Data-driven methods in machine translation*, Toulouse, France.

[12] Moore, Robert C. (2002) Fast and Accurate Sentence Alignment of Bilingual Corpora. In *Proceedings of the 5th Conference of the Association for Machine Translation in the Americas: From Research to Real Users*, Tiburon, California.

[13] Samuelsson, Yvonne, Volk, Martin, Gustafson–Capková, Sofia and Steiner, Elisabet Jönsson (2010) *Alignment Guidelines for SMULTRON*. Stockholm University, Department of Linguistics, University of Zürich, Institute of Computational Linguistics. Version 2.1, August 9, 2010.

[14] Simov, K., Osenova, P., Laskova, L., Savkov, A. and Kancheva, S. (2011) Bulgarian-English Parallel Treebank: Word and Semantic Level Alignment. In *Proceedings of Recent Advances in Natural Language Processing* (RANLP), Hissar, Bulgaria.

[15] Steinberger, R. and Pouliquen, B. and Widiger, A. and Ignat, C. and Erjavec, T. and Tufiş, D. and Varga, D. (2006) The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages. In *Proceedings of Language Resources and Evaluation Conference* (LREC'06), Genova, Italy.

[16] Zhechev, Ventislav and Way, Andy (2008) Automatic generation of parallel treebanks. In *22nd International Conference on Computational Linguistics* (COLING 2008), Manchester, UK.

Building a treebank of noisy user-generated content: The French Social Media Bank

Djamé Seddah Benoît Sagot Marie Candito
Virginie Mouilleron Vanessa Combet
Alpage, Inria Paris-Rocquencourt & Université Paris Diderot, France

Abstract

We introduce the French Social Media Bank, the first user-generated content treebank for French. Its first release contains 1,700 sentences from various Web 2.0 and social media sources (FACEBOOK, TWITTER, web forums), including data specifically chosen for their high noisiness.

1 Introduction

New forms of electronic communication have emerged in the last few years, namely social media and Web 2.0 communication media, both synchronous (e.g., micro-blogging) or asynchronous (e.g., forums). These new user-generated contents often depart, sometimes heavily, from canonical language. This prevents an accurate processing of such data by current state-of-art NLP tools (Foster et al. [9], Gimpel et al. [11]). The main difficulties, highlighted by Foster [8], range from surface differences (intended or accidental non-standard typography) to lexical idiosyncrasies (genuine unknown words, sloppy spelling) and specific syntactic structures absent from well-edited data (imperatives, direct speech, slang, etc.).

The still difficult handling of those phenomena pleads for a better linguistic modeling and analysis of user-generated content. We introduce the French Social Media Bank, a freely available treebank containing 1700 manually annotated sentences. It constitutes the first resource covering the variety of French social medias, and the first data set we are aware of for FACEBOOK data.

2 Corpus

The French web 2.0 covers a wide range of practices. We decided to focus on microblogging (FACEBOOK and TWITTER) and on two types of web forums: one large-audience health forum, DOCTISSIMO (forum.doctissimo.fr) and one specialized on video games JEUXVIDEOS.COM (www.jeuxvideo.com). For each source but the latter, we gathered both lightly edited data and *noisier* data, using handcrafted search queries. Lightly edited data were retrieved based on source-specific news topics. The noisiest texts, intended to serve as a stress test for

	# sent.	# tokens	avg. lgth	std dev.	noisiness score
DOCTISSIMO	771	10834	14.05	10.28	0.37
high noisiness subcorpora	36	640	17.78	17.63	1.29
other subcorpora	735	10194	13.87	9.74	0.31
JEUXVIDEOS.COM	199	3058	15.37	14.44	0.81
TWITTER	216	2465	11.41	7.81	1.24
high noisiness subcorpora	93	1126	12.11	8.51	1.46
other subcorpora	123	1339	10.89	7.20	1.08
FACEBOOK	452	4200	9.29	8.17	1.67
high noisiness subcorpora	120	1012	8.43	7.12	2.44
other subcorpora	332	3188	9.60	8.49	1.30

Table 1: Corpus properties

French linguistic modeling and statistical parsing, were obtained by looking for slang words and urban idiomatic constructions. Table 1 presents some properties of our corpora.

In order to quantitatively assess the level of noisiness in our corpora we defined an ad-hoc *noisiness* metric. It is defined as a variant of the Kullback–Leibler divergence between the distribution of trigrams of characters in a given corpus and that in a reference corpus (in our case, the French Treebank (Abeillé et al. [1]), hereafter FTB). The figures given in Table 1 are consistent with our classification in two noisiness levels. We used this metric to decide for each sub-corpus whether to apply a standard pre-annotation or a dedicated noise-tolerant architecture instead (cf. Section 5).

We refer the reader to (Seddah et al. [13]) for more detail on our various subcorpora. We provide here two examples, (1) from the lightly edited TWITTER subcorpus, and (2), from the our high-noisiness FACEBOOK subcorpus.

(1) Je soupçonnes que "l'enfarineuse" était en faite une cocaineuse vu la pêche de #Hollande ce soir à #Rouen.
Je soupçonne que l'enfarineuse était en fait une cocaïneuse vu la pêche de #Hollande ce soir à #Rouen.
I suspect that the "flouring-lady" was actually a cocaïn-lady given the energy of #Hollande this night at #Rouen.

(2) L'Ange Michael vraiment super conten pour toi mé tora plus grace a moi tkt love you!
L'Ange Michael: (Je suis) Vraiment super content pour mais tu auras plus grace à moi. Ne t'inquiètes pas. Je t'aime !
The Angel Mickael: (I am) Really very happy for him but you'll get more because of me. Don't worry. I love you!

3 Linguistics of user generated content

User-generated texts do not correspond to a single homogenous domain, although some specificities of user-generated content are found across various types of web data. Moreover, in some cases, and most notably TWITTER, such data include both linguistic content and media-specific meta-language. This meta-language (such as

Phenomenon	Attested example	Std. counterpart	Gloss
	ERGOGRAPHIC PHENOMENA		
Diacritic removal	*demain c'est l'***ete**	*demain c'est l'été*	'tomorrow is summer'
Phonetization	*je suis* **oqp**	*je suis occupé*	'I'm busy'
Simplification	*je* **sé**	*je sais*	'I know'
Spelling errors	*tous mes* **examen** **son** *normaux*	*tous mes examens sont normaux*	'All my examinations are normal'
	TRANSVERSE PHENOMENA		
Contraction	**nimp**	*n'importe quoi*	'rubbish'
	qil	*qu'il*	'that he'
Typographic diaeresis	**c a dire**	*c'est-à-dire*	'namely'
	c t	*c'était*	'it was'
	MARKS OF EXPRESSIVENESS		
Punctuation transgression	*Joli !!!!!!*	*Joli !*	'nice!'
raphemic stretching	*superrrrrrrrr*	*super*	'great'
Emoticons/smileys	*:-)*, <3	–	–

Table 2: A few idiosyncrasies found within French user-generated content

TWITTER's "RT" ("Retweet"), at-mentions and hashtags) is to be extracted before parsing *per se* or other types of linguistic processing. In this work, we focus on the linguistic content. Therefore, we deal with meta-language tokens only when they are embedded within or adjacent to purely linguistic content (e.g., the tweet itself, provided it consists of one or several sentences).

Prevalent idiosyncrasies in user generated content can be characterized on two axes: one which can be roughly describe as "the encoding simplification axis" which covers ergographic[1] and transverse phenomena and the other "sentiment expression axis" which covers phenomena, or marks of expressiveness, that emulate the same goal as sentiment expressed through prosody and gesture in direct interaction. Table 2 gathers the most striking of these phenomena.

These artifacts lead to a high unknown word level. More importantly, the new morphology brought by the those phenomenon complicates any suffix-based unknown word analysis. Nevertheless, our general annotation strategy consists in staying as consistent as possible with the FTB guidelines (Abeillé et al. [1]).

4 Annotation scheme

We followed the FTB annotation guidelines (Abeillé et al. [1]). More precisely, we based our annotation scheme on its FTB-UC variant (Candito and Crabbé [3]) which was optimized for parsing purposes. It mainly departs from the original FTB on the tagset granularity and on the modeling of multiword units. We added specific guidelines to handle idiosyncrasies user-generated content corpora.

We also added two new POS tags, namely *HT* for TWITTER hashtags and *META* for meta-textual tokens, such as TWITTER "RT". TWITTER at-mentions as well as URLs and e-mail addresses have been tagged *NPP*. The rationale for

[1] Phenomenon aiming at reducing the writing effort.

this is to remain consistent with our tagging and parsing models trained on the FTB, which do not contain such tokens. This constitutes the main difference with other works on user-generated data (Gimpel et al. [11]). One other major extension at the POS level concerns contraction and typographic diaeresis phenomena (see Section 3). Contracted tokens are associated with a combined POS tag which lists the sequence of each underlying words' tag. Let us consider for example, the non-standard contraction *jai*, which stands for *j' ai*, which would have been tagged *CLS* and *V* (subject clitic and finite verb). The non-standard contracted token *jai* is then tagged *CLS+V*. In this case, the contraction involves a verb and one of its argument. In such situations, function labels are associated directly with the contracted token. For cases of typographic diaeresis, the category of its standard counterpart is given to the last token, all others receive the special tag *Y*. For example, *c a dire* stands for the conjunction *c'est-à-dire*, which would have been tagged *CC*. We thus tag the first two tokens as *Y* and *dire* as *CC*. This is consistent with how such cases are handled in the English Web Treebank (Bies et al. [2]).

At the syntactic level, the main addition to the FTB-UC tagset is a new FRAG label, for phrases that cannot be syntactically attached to the main clause of a syntactic unit (e.g., salutations, emoticons...). It also covers usernames, at-mentions, and URL appended to a sentence.

These extensions are largely compatible with the English Web Bank. However, our treebank differs from the former in several aspects. First, French has a richer morphology than English, entailing a tedious disambiguation process when facing *noisy* data. Although the first version of our treebank is smaller than the English Web Treebank, it includes richer annotations (compound POS, corrected token form of contractions) and includes subcorpora exhibiting a very high level of noise.

5 Annotation Methodology

We built our manually validated treebank following a well established methodology: we first defined a sequence of annotation layers, namely (i) sentence splitting, tokenization and POS tagging, (ii) syntagmatic parsing, (iii) functional annotation. Each layer is annotated by an automatic preprocessing that relies on previously annotated layers, followed by validation and correction by human annotators. At each step, annotators were able to modify choices made at previous stages.

We used two different strategies for tokenization and POS pre-annotation of our sub-corpora, depending on their noisiness score. For less *noisy* corpora (noisiness score below 1), we used a slightly extended version of the tokenization tools from the FTB-based parsing architecture Bonsai (Candito et al. [4]), in order to match as much as possible the FTB's tokenization scheme. Next, we used the POS-tagger MORFETTE (Chrupała et al. [5]). For corpora with a high noisiness score, we used a specifically developped pre-annotation process. This is because in such corpora, spelling errors are even more frequent, but also because the original tokens rarely match sound linguistic units. The idea underlying this pre-processing is to wrap

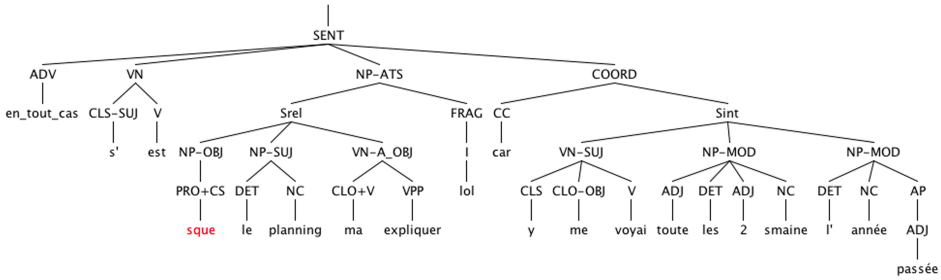


Figure 1: French Social Media Bank's sample of the *noisy*DOCTISSIMO subcorpus. English gloss: 'Anyway that's what the social centre explained to me lol he was actually seeing me every two weeks last year.'

the POS tagger (in this case, MElt, (Denis and Sagot [7])) within a temporary text normalization tool, so that the tagger is provided with data as close as possible to its training corpus, the FTB.

Parse pre-annotation was achieved using a state-of-the-art statistical parser trained on the FTB-UC, provided with manually corrected POS tags. We used the Berkeley parser (Petrov et al. [12]) adapted to French (Crabbé and Candito [6]). Note that when the validated POS tags were discarded by the parser, in case of too many unknown word-POS pairs, those were reinserted. Functional annotation was carried out as a post-parsing stage using the associated labeler (Candito et al. [4]) and then manually validated. An example of the resulting annotation is shown Figure 1.

6 Conclusion

The French Social Media Bank shares with the English Web Treebank (Bies et al. [2]) a common will to extend the treebank domain towards user generated content. Although of a smaller scale, it constitutes one of the very first resources for validating social media parsing and POS tagging, together with DCU's Twitter & BBC football forum treebanks (Foster et al. [9, 10]) and the Twitter POS data set from Gimpel et al. [11]. Moreover, it is the first set of syntactically annotated FACEBOOK data and the first treebank of its kind for French.

We performed a first round of evaluation showing that simple techniques could be used to improve POS tagging performance. Indeed, raw accuracy results of the MElt POS-tagger, which gives state-of-the-art results on edited texts, range from 56 % (DOCTISSIMO-noisy) to 87 % (TWITTER), whereas the use of the dedicated wrapper mentioned in Section 5 leads to figures between 80 % and 89 %. We have also achieved baseline statistical parsing results, with results far behind those on newspaper in-domain texts (Evalb's f-measures ranging from 39 % to 70 %, to be compared with 86–89 % regularly achieved on the FTB test set). These preliminary results prove the difficulty of processing such data and therefore the importance of building a data set such as the French Social Media Bank.

Acknowledgments This work was partly funded by the French ANR project EDyLex (ANR-09-CORD-008).

References

[1] Abeillé, A., Clément, L., and Toussenel, F. (2003). *Building a Treebank for French*. Kluwer, Dordrecht.

[2] Bies, A., Mott, J., Warner, C., and Kulick, S. (2012). English web treebank. Technical report, Linguistic Data Consortium,, Philadelphia, PA, USA.

[3] Candito, M. and Crabbé, B. (2009). Improving generative statistical parsing with semi-supervised word clustering. In *Proc. of IWPT'09*, Paris, France.

[4] Candito, M., Nivre, J., Denis, P., and Henestroza, E. (2010). Benchmarking of statistical dependency parsers for french. In *Proc. of CoLing'10*, Beijing, China.

[5] Chrupała, G., Dinu, G., and van Genabith, J. (2008). Learning morphology with morfette. In *In Proceedings of LREC 2008*, Marrakech, Morocco.

[6] Crabbé, B. and Candito, M. (2008). Expériences d'analyse syntaxique statistique du français. In *Proc. of TALN'08*, pages 45–54, Senlis, France.

[7] Denis, P. and Sagot, B. (2009). Coupling an annotated corpus and a morphosyntactic lexicon for state-of-the-art POS tagging with less human effort. In *Proc. of PACLIC*, Hong Kong, China.

[8] Foster, J. (2010). "cba to check the spelling": Investigating parser performance on discussion forum posts. In *Proc. of HLT/NAACL'10*, Los Angeles, USA.

[9] Foster, J., Cetinoglu, O., Wagner, J., Le Roux, J., Hogan, S., Nivre, J., Hogan, D., and van Genabith, J. (2011a). #hardtoparse: Pos tagging and parsing the twitterverse. In *Proc. of the AAAI 2011 Workshop On Analyzing Microtext*.

[10] Foster, J., Cetinoglu, O., Wagner, J., Le Roux, J., Nivre, J., Hogan, D., and van Genabith, J. (2011b). From news to comment: Resources and benchmarks for parsing the language of web 2.0. In *proc of IJCNLP*, Chiang Mai, Thailand.

[11] Gimpel, K., Schneider, N., O'Connor, B., Das, D., Mills, D., Eisenstein, J., Heilman, M., Yogatama, D., Flanigan, J., and Smith, N. A. (2011). Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proc. of ACL'11*, Portland, USA.

[12] Petrov, S., Barrett, L., Thibaux, R., and Klein, D. (2006). Learning accurate, compact, and interpretable tree annotation. In *Proc. of ACL'06*, Sydney, Australia.

[13] Seddah, D., Sagot, B., Candito, M., Mouilleron, V., and Combet, V. (2012). The french social media bank: a treebank of noisy user generated content. In *Proceedings of CoLing'12*, Mumbai, India.

Impact of treebank characteristics on cross-lingual parser adaptation

Arne Skjærholt, Lilja Øvrelid

Department of informatics, University of Oslo
{arnskj,liljao}@ifi.uio.no

Abstract

Treebank creation can benefit from the use of a parser. Recent work on cross-lingual parser adaptation has presented results that make this a viable option as an early-stage preprocessor in cases where there are no (or not enough) resources available to train a statistical parser for a language.

In this paper we examine cross-lingual parser adaptation between three highly related languages: Swedish, Danish and Norwegian. Our focus on related languages allows for an in-depth study of factors influencing the performance of the adapted parsers and we examine the influence of annotation strategy and treebank size. Our results show that a simple conversion process can give very good results, and with a few simple, linguistically informed, changes to the source data, even better results can be obtained, even with a source treebank that is quite differently annotated from the target treebank. We also show that for languages with large amounts of lexical overlap, delexicalisation is not necessary, indeed lexicalised parsers outperform delexicalised parsers, and that for some treebanks it is possible to convert dependency relations and create a *labelled* cross-lingual parser.[1]

1 Introduction

It is well known that when annotating new resources, the best way to go about it is not necessarily to annotate raw data, but rather to correct automatically annotated material (Chiou et al. [2], Fort and Sagot [3]). This can yield important speed gains, and often also improvements in measures of annotation quality such as inter-annotator agreement. However, if we wish to create a treebank for a language with no pre-existing resources, we face something of a Catch-22; on the one hand, we have no data to train a statistical parser to automatically annotate sentences, but on the other hand we really would like to have one. In this setting, one option is cross-lingual parser adaptation.

[1] The code used to obtain the data in this paper are available from `https://github.com/arnsholt/tlt11-experiments`.

Cross-lingual parser adaptation has previously been proposed both for closely related source-target language pairs (Zeman and Resnik [12]) and less related languages (Søgaard [9], Täckström et al. [10]). The basic procedure has been very similar in all of this work: a simple conversion procedure is applied to map the part-of-speech tags of the source and target languages into a common tagset and a delexicalised parser is subsequently trained on (a possibly filtered version of) the source treebank and applied to the target language.

The choice of language to use as a starting point for adaptation might not be obvious. There can be several candidates, each with different pros and cons, or there may be no obvious candidates, and the question is rather which choice will be the least bad. The results obtained in previous work have varied quite a bit and it is clear that several factors influence this type of parsing. First of all, languages may differ with respect to typological properties, such as word order, the role of morphology, etc., making them more or less apt for this type of direct transfer of linguistic structure. Regardless of degree of linguistic relatedness, treebanks may also implement different annotation strategies, which will at times require quite sophisticated conversion procedures for proper evaluation. Finally, the treebanks used for parser adaptation in previous work have varied distinctly in terms of size. It is at this point unclear how these different factors influence results.

In this paper, we investigate cross-lingual parser adaptation between highly related languages, namely the Scandinavian languages of Swedish, Danish and Norwegian. An ongoing effort to produce a dependency treebank for Norwegian enables an experimental setting where the degree of relatedness between source and target language can be kept fairly constant and we can examine the effect of differing annotation strategies and source treebank size on cross-lingual parser adaptation results. In this paper we present experiments that attempt to disentangle these factors and furthermore show how linguistic relatedness allows us to relax the requirement for delexicalisation assumed in previous work.

1.1 Related work

Zeman and Resnik [12] first reported on experiments with parser adaptation between related languages (Swedish and Danish), where they map the treebanks to a common tagset and perform some structural conversions in order to make the annotations more similar. Prior to training, both treebanks are converted to phrase structure representations and Charniak and Johnson's reranking parser (Charniak and Johnson [1]) is used for the parsing experiments. They show that this technique can be useful in a first stage of treebanking, and report best results that are equivalent to training with approximately 1500 target language instances.

Søgaard [9] extends the approach of Zeman and Resnik [12] to unrelated languages (Arabic, Danish, Bulgarian and Portugese). He introduces a filtering step where the source treebank is modified to resemble the target treebank. The filtering is performed using a language model trained over target language part-of-speech sequences, which is then used to filter out sentences in the source language tree-

(a) Danish (b) Swedish/Norwegian

Figure 1: Noun phrase annotation

bank that fall below a certain perplexity threshold. All experiments are performed with dependency representations, using MSTparser (McDonald et al. [5]). The work shows that the filtering approach yields considerable improvements over the simple technique proposed by Zeman and Resnik [12] when applied to unrelated languages.

The above approaches to parser adaptation train delexicalized parsers on the (unfiltered or filtered) source language data, however, there are clearly many lexical patterns that are similar cross-lingually if one finds the right level of abstraction. Täckström et al. [10] investigate the use of cross-lingual word clusters derived using parallel texts in order to add some level of lexical information that abstracts over individual words.

2 Treebanks

In all our experiments, the target language is Norwegian, and either Swedish or Danish is used as the source. For these experiments, we used three treebanks: the Danish Dependency Treebank (DDT) (Kromann [4]) and the Swedish Talbanken05 (Nivre et al. [6]), both from the 2006 CoNLL shared task on multilingual dependency parsing, and an in-development Norwegian treebank being prepared by the Norwegian national library (NDT)[2]. Of the three, Talbanken is by far the largest, with some 11000 sentences in the training set. The DDT is smaller with about 5200 sentences of training data, and the NDT even smaller with 4400 sentences all told.

The Scandinavian languages, Danish, Norwegian and Swedish, are extremely closely related to one another. The split of Norse into the modern-day Scandinavian languages is very recent[3], which means that the languages are mutually intelligible to a large extent, and share a large number of syntactic and morphological features, and have a large common vocabulary. Perhaps the largest difference is orthographical conventions.

Our treebanks are more diverse than the languages they are based on. As Zeman and Resnik [12] point out, Talbanken and DDT are different in many respects; most of these differences are due to DDT preferring functional heads while Tal-

[2]A beta release is available at `http://www.nb.no/spraakbanken/tilgjengelege-ressursar/tekstressursar`

[3]Within the last 1000 to 500 years. For comparison, the split between German and Dutch is some 500 years before this.

(a) Danish (b) Swedish/Norwegian

Figure 2: Subordinate clause annotation

banken to a larger extent prefers lexical heads. NDT largely follows the Talbanken annotation strategy with some minor exceptions. During the development of the Norwegian treebank, Talbanken was an important reference point, and as a result the two treebanks are very similar. The functional focus of DDT is apparent in the treatment of noun phrases, where determiners instead of nouns are heads, as illustrated by Figure 1. The functional analysis is also found in subordinate clauses, illustrated by Figure 2 where DDT treats the subjunction as head of the subordinated verb, whereas the analysis in the other two treebanks appoints the finite verb to be the head. This means that important constructions, such as noun phrases and subordinate clauses have very dissimilar analyses.

3 Experiments

Even though the level of linguistic relatedness is more or less constant for the NDT-Talbanken and NDT-DDT corpus pairs, there are a number of other parameters that can vary; for our present work we focus on the impact of annotation strategy used by the source treebank, the importance of size of the source treebank, and the utility of converting the source data to closer resemble the target representation. For all our experiments, we use the CoNLL 2006 training sets to train the respective parsers, and the final 10% (442 sentences) of the NDT as the evaluation set. For the sake of brevity, we use the abbreviations *no-sv* and *no-da* for Norwegian parsed by a parser trained on Swedish or Danish data, respectively.

For all our parsers, we use the same parser setup. Models are trained with MaltParser (Nivre et al. [7]) using the nivreeager parsing algorithm and the default feature model. Initial experiments (using Talbanken) had the nivreeager algorithm consistently outperforming the other algorithms, and to keep the results of our various experiments as comparable as possible, we use a single parser setting for all the experiments.

3.1 Basic strategy

Our first set of experiments set out to explore the possibilities of an approach that involves as little effort as possible. Following Zeman and Resnik [12] and Søgaard [9] we map the PoS tags of all three treebanks to a common tagset (similar to Petrov et al.'s [8] universal tagset) using Daniel Zeman's interset (Zeman [11]) and a new

sv:	Bestämmelserna	i	detta	avtal	får	ändras	eller	revideras
da:	Bestemmelserne	i	denne	aftale	kan	ændres	og	revideres
no:	Bestemmelsene	i	denne	avtalen	kan	endres	eller	revideres

helt	eller	delvis	efter	gemensam	överenskomst	mellan	parterna.
helt	eller	delvis	efter	fælles	overenskomst	mellem	parterne.
helt	eller	delvis	etter	felles	overenskomst	mellom	partene.

en: The provisions of the Agreement may be amended and revised as a whole or in detail by common agreement between the two Parties.

Figure 3: Swedish-Danish parallel example.

driver we have written for the Norwegian tagset. We then delexicalise the corpora, replacing the word form and lemma columns in the CoNLL data with dummy values, leaving only the PoS information in the treebank, and train parsers on the delexicalised treebanks. The "Basic" column of the first two rows of Table 1 show the performance of these parsers on the Norwegian test corpus. We tried to apply Søgaard's [9] filtering technique in conjunction with these experiments, but it did not have any noticeable impact on performance, confirming Søgaard's hypothesis that this technique is most useful for unrelated or distantly related languages.

3.2 Lexicalised parsing

Not delexicalising the source corpora in a cross-lingual setting is unconventional, but as noted above the Scandinavian languages are extremely similar. Figure 3, shows an example parallel sentence[4] (Zeman and Resnik [12]), and we may note that of the 16 words in the sentence, 5 are identical between the Danish and Swedish versions. In the Norwegian version, 9 words (*i*, *kan*, *og*, *revideres*, *helt*, *eller*, *delvis*, *overenskomst*) are indentical to the word in the Danish version and 6 (*i*, *får*, *eller* [twice], *helt*, *delvis*) in the Swedish. Given this similarity, our hypothesis was that keeping the words might give a small performance boost by making it possible to learn some selectional preferences during parser training.

Looking more closely at the lexical overlaps between the training corpora and our test corpus, we find that, disregarding punctuation, out of the word-forms encountered in the Swedish training set, 338 also occur in the test corpus (a total of 3151 times), and 660 word-forms from the Danish corpus occur a total of 4273 times. This means that out of the 6809 non-punctuation tokens in the Norwegian test corpus, 46% of them are known words to a lexicalised Swedish parser and 63% are known to a lexicalised Danish parser.

The "Basic" column of Table 1 shows the results of our experiments with the parsers described in this and the previous section. As the table makes clear, Talbanken fares far better than DDT as source corpus. Given the large overlap in

[4]The sentence is from the Acquis corpus. The Swedish, Danish and English versions are those in Acquis; Norway not being a member of the EU, we have supplied our own Norwegian translation.

Configuration	Basic	Converted	
		UAS	LAS
no-sv, delexicalised	66.8%	73.9%	64.0%
no-da, delexicalised	46.3%	72.5%	39.2%
no-sv, lexicalised	75.5%	79.1%	68.7%
no-da, lexicalised	52.4%	74.2%	43.3%
Norwegian skyline	—	87.4%	84.6%

Table 1: Parser performances. Basic configuration only unlabelled attachment, Converted configuration both unlabelled (UAS) and labelled (LAS) attachment scores.

terms of annotation strategy, this is not surprising. Also, it is clear that delexicalisation is not necessary for languages as closely related as Norwegian and Danish or Swedish. The boost in performance from a lexicalised parser is far bigger for Swedish than Danish, contrary to what we might assume based on the lexical overlap figures, but as we shall see in Section 3.4, this is not necessarily all due to the lexical overlaps; the important differences in annotation strategy between DDT and NDT is likely a factor as well.

3.3 Size

Given that the Talbanken corpus is roughly twice the size of the DDT, it is a relevant question whether the difference between the two in the results of the previous experiments is due to size. To explore this, we plotted learning curves for all four parser configurations from the previous section; as a skyline on the possible performance, we used the 90% of the NDT not used for testing. We trained the parsers by taking the first n sentences of the relevant corpus, stepping n roughly exponentially[5] until the whole treebank was used.

As the learning curves in figure 4 make clear, corpus size is not an important factor here. The curves for Danish and Swedish show much the same behaviour: the first few sentences do relatively little for performance up to the 10 sentence mark, at which point performance increases more rapidly. When the number of sentences reaches the mid-hundreds, the improvements start to flatten out, and beyond the 1000 sentence mark the delexicalised parsers start to deteriorate, most likely due to over-training.

The learning curves for the cross-lingual parsers show approximately the same evolution as the upper bound established by the parser trained directly on Norwegian data, but the increase in performance as the amount of data increases is less pronounced for the parser trained on Swedish, and the one trained on Danish is almost flat.

[5] $n \in \{10, 20, 50, 100, 200, 500, 1000 \ldots\}$

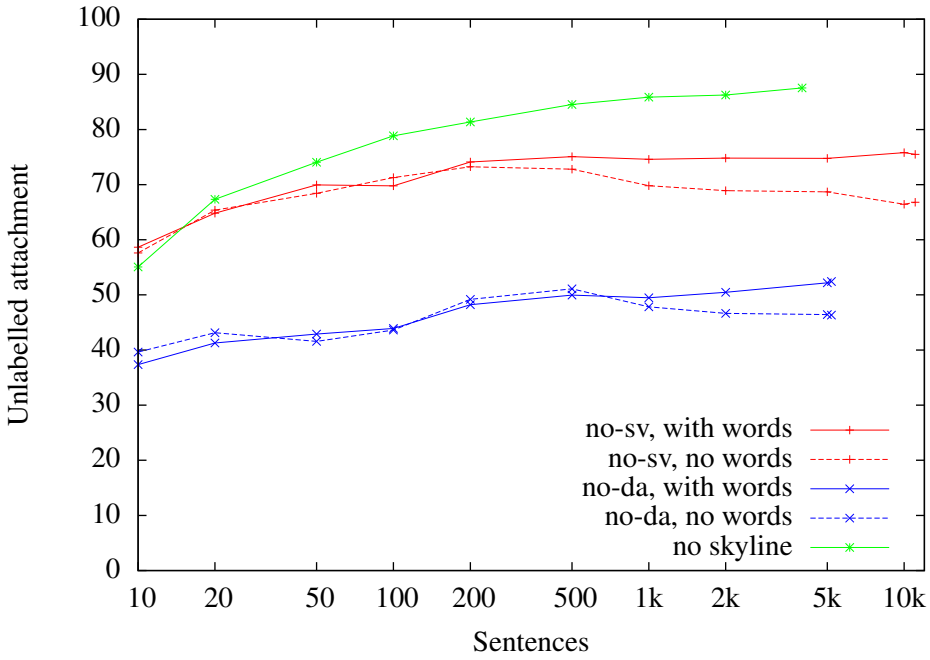


Figure 4: Learning curves for parsers using the Basic strategy

3.4 Conversion

As outlined in Section 2 there are a number of differences in terms of annotation between our three corpora, in particular between the DDT and Talbanken/NDT. In this section we present a set of more specific conversion procedures we use to move our source corpora closer to the target corpus. Previous work in cross-lingual parser adaptation (Zeman and Resnik [12], Søgaard [9]) has considered only unlabelled parsing, but as we will see in Section 3.4.3, it is entirely possible to make a labelling parser.

3.4.1 Part-of-speech conversion

Until now, only the bare minimum of conversion and mapping has been applied to the source corpora. But many of the differences in annotation strategy between the source and target corpora are quite simple and easy to recover, and given the close connection between the languages, it is not hard to do a more targeted PoS tag mapping than what interset provides.

Specifically, we can convert both Talbanken and DDT's PoS tagsets into the tagset used by the NDT. For the most part, this is a simple matter of writing down a look-up table mapping the tags in the source tagset to the corresponding tag in the target tagset. Roughly 90% of both the Swedish and Danish tags can be converted

(a) Swedish (b) Norwegian

Figure 5: Infinitival clause annotation

in this manner; the remaining require special logic. For example, the Swedish tag TP (perfect participle) is variously tagged as adjective (adj) or verb (verb) in the NDT, depending on syntactic function. Swedish participles that are annotated as determiners (DT), adjectival modifiers (AT) or arguments of prepositions (PA) are then mapped to the Norwegian adjective tag and participles with other dependency relations receive the verb tag.

3.4.2 Structural conversion

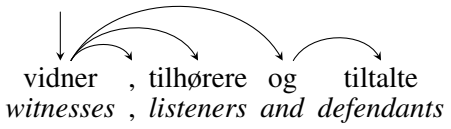
In the same vein, the majority of the differences in annotation strategy between the treebanks are systematic differences. As previously noted, Talbanken and NDT are very similar, but they do differ systematically in their analyses of infinitival phrases, as shown in Figure 5; where Talbanken has the infinitive as the head and the infinitive marker as a dependent of the verb, NDT has the infinitive marker as the head of the whole construction.

Given the preference for functional over lexical analyses in DDT, more structural conversions are required when converting the DDT to NDT's format. We convert a number of structures, all of them in essence due to functional heads. In coordination, illustrated in Figure 6, DDT has the coordinator as a dependent of the first conjunct and the final conjunct depending on the coordinator, while NDT inverts the relationship between the final conjunct and the coordinator, so that the coordinator depends on the final conjunct, which is attached to the first conjunct.

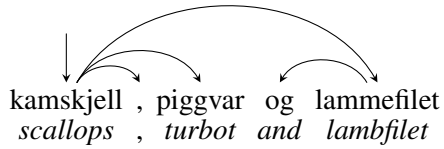
Two more structural conversions are due to the analysis of noun phrases show in Figure 1: Genitive-Nominative and Determiner-Noun constructions. The Genitive-Noun case simply inverts the relationship between Genitive (the head in the DDT analysis) and the Nominative in the same way as we did with the coordinator and the final conjunct in coordination. The relationship between determiner and noun is inverted in Determiner-Noun constructions, but any other dependents of the determiner are also made into dependents of the noun, since the DDT will have any modifiers as dependents of the determiner rather than the noun[6].

All of these differences are also converted by Zeman and Resnik [12]. However, we expand on their procedure by also dealing with subordination. As shown

[6]Zeman and Resnik [12] separate Det-Adj-Noun and Num-Noun as separate cases, but our code handles both cases in a single code-path.



(a) Danish



(b) Norwegian

Figure 6: Coordination structures

in Figure 2, DDT and NDT have diverging analyses of subordination, which is straightforwardly handled in the same way as the Determiner-Noun case: the subordinator is made a dependent of the finite verb, and any further children of the subordinator are also moved to the verb.

3.4.3 Dependency relation conversion

Our dependency relation conversion procedure operates in essentially the same way as our PoS mappings: mostly lookup tables converting one relation to an NDT equivalent, with some additional logic to handle the remaining harder cases. Most of the special handling involves checking the PoS tag on the head or the word, like the Swedish tags +A, RA and TA (conjunctional, place and time adverbials) which are labelled ATR (attributive) when the head is nominal and ADV (adverbial) when the head is verbal.

Converting the Danish dependency relations is not qualitatively different from the Swedish relations, however many of the distinctions made by the DDT dependency relations are orthogonal to the distinctions made by the NDT, which complicates the conversion process. In particular relations like aobj, nobj and vobj (adjectival, nominal and verbal objects) were non-trivial to map. Inspecting the head, and in the case of the relative subjunction the actual form of the token, helps to some extent, but in the end the Danish mapping procedure was less successful than the Swedish one. This is also reflected in the increased number of special cases: 87% of the Swedish relations are handled via look-ups, but only 66% of the Danish ones are.

3.4.4 Results

Results for our parsers trained on the converted treebanks are shown in the "Converted" columns of Table 1. Conversion obviously helps, giving large improvements in performance for both the lexicalised and delexicalised parsers. The Danish parser has far more to gain from the conversion process than the Swedish parser, with both the delexicalised and lexicalised parsers increasing by more than 20 percentage points unlabelled accuracy.

The labelled attachment scores of the converted parsers really bring out the difference between the conversion of the Swedish dependency relations and that of the Danish relations: The difference between labelled and unlabelled attachment is about 10 percentage points for both the lexicalised and delexicalised Swedish parsers, while the Danish parsers have gap of 30 points between labelled and unlabelled.

An interesting observation is that the conversion process is more useful for the delexicalised parsers than the lexicalised ones. The Swedish delexicalised parser gains 7.1 percentage points (of unlabelled attachment) from the conversion process, while the lexicalised one only gains 3.6 points, and the delexicalised Danish parser gains more than 25 percentage points, while the lexicalised parser gains 22 points. We take this as further evidence that lexicalisation is useful for these closely related languages: The known words give the parser some knowledge about selectional preferences for some words, which means that the improvement in coarse-grained generalisations offered by the more targeted PoS conversion has less of an impact.

4 Conclusions

From these experimental results, we can make a number of practical conclusions on when and where cross-lingual parsing is a viable approach, as well as how to go about it. If a treebank is available for a closely related language with a similar annotation strategy as that used by a new resource, a decent parser can be obtained with almost no work at all; all that is required is a script to map the source corpus PoS tags with interset to train a parser.

But with a quite modest investment of time (defining and implementing the conversions used for our work did not take more than a day or two divided between the two authors) important gains in performance can be had, especially if the candidate source treebank that is the closest linguistically to the target language has made diverging choices of analysis of important structures. That this process is quick is rather important, as once the number of annotated sentences reaches the mid-to-high hundreds, it is likely that a parser trained on already annotated material will outperform a ported parser, and as such it is best to not spend too much time on the adaptation process itself.

The effort required to define and implement dependency relation mappings is roughly the same as that required for a targeted PoS tag mapping, and if the relation sets of the target and source treebanks align well enough the labelled parser perfor-

mance is not inconsiderable, as was the case for Talbanken and NDT. However, as evidenced by the DDT-NDT pair, the labelled performance varies more in function of the characteristics of the source treebank.

Finally, it turns out that delexicalisation is not required if the languages in question are closely enough related that there is non-trivial amounts of lexical overlap between the languages. But if a delexicalised parser is required and the source treebank is large, it may be best to not use the entire treebank to train the adapted parser. As the learning curves in Figure 4 show, a delexicalised parser trained on more than 500-1,000 sentences starts to overtrain and is likely to perform worse than a parser trained on less material on target language data.

5 Future work

There are a number of topics that merit further investigation. First of all, a thorough study is required of the influence of using pre-parsed data for annotation and the influence of parser performance on the parameters of annotation process, such as time used per sentence and inter-annotator agreement. Fort and Sagot [3] study the influence of pre-processed data for the part-of-speech annotation task, and Chiou et al. [2] show that an accurate parser speeds up phrase structure annotation, but to our knowledge no such study has been done for the task of dependency annotation.

Given languages as similar as the languages used here, it would also be quite interesting to discard the notion of three distinct languages, and rather consider them three dialects of the same language and apply orthographic normalisation techniques to automatically convert Swedish or Danish data to Norwegian.

Finally comes the question of parameter tuning. In an in-domain setting techniques such as cross-validation over the training set are useful techniques to find the best set of parameters for an algorithm on a data-set, but for cross-lingual parsing, it is not at all obvious that the model that performs optimally on the source-language data is the best model for the target domain. It is thus necessary to further investigate the possible correlation between source and target domain performance and methods of *a priori* estimation of performance on out-of-domain data.

References

[1] Charniak, E. and Johnson, M. (2005). Coarse-to-Fine N-Best Parsing and MaxEnt Discriminative Reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 173–180, Ann Arbor, Michigan. Association for Computational Linguistics.

[2] Chiou, F.-D., Chiang, D., and Palmer, M. (2001). Facilitating Treebank Annotation Using a Statistical Parser. In *Proceedings of the first international conference on Human language technology research*, pages 1–4.

[3] Fort, K. and Sagot, B. (2010). Influence of Pre-annotation on POS-tagged Corpus Development. In Xue, N. and Poesio, M., editors, *Proceedings of the fourth linguistic annotation workshop*, pages 56–63, Stroudsburg. Association for Computational Linguistics.

[4] Kromann, M. T. (2003). The Danish Dependency Treebank and the DTAG treebank tool. In *Proceedings from the Second Workshop on Treebanks and Linguistic Theories (TLT 2003)*.

[5] McDonald, R., Crammer, K., and Pereira, F. (2005). Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 91–98, Morristown, NJ, USA. Association for Computational Linguistics.

[6] Nivre, J., Nilsson, J., and Hall, J. (2006). Talbanken05: A Swedish treebank with phrase structure and dependency annotation. In *Proceedings of LREC*, pages 1392–1395.

[7] Nivre, J., Nilsson, J., Hall, J., Chanev, A., Eryiit, G., Kübler, S., Marinov, S., and Marsi, E. (2007). MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

[8] Petrov, S., Das, D., and McDonald, R. (2011). A Universal Part-of-Speech Tagset.

[9] Søgaard, A. (2011). Data point selection for cross-language adaptation of dependency parsers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 682–686.

[10] Täckström, O., McDonald, R., and Uszkoreit, J. (2012). Cross-lingual Word Clusters for Direct Transfer of Linguistic Structure. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 477–487. Association for Computational Linguistics.

[11] Zeman, D. (2008). Reusable Tagset Conversion Using Tagset Drivers. In *Proceedings of the Language Resources and Evaluation Conference 2008*.

[12] Zeman, D. and Resnik, P. (2008). Cross-Language Parser Adaptation between Related Languages. In Singh, A. K., editor, *Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages*, pages 35–42, Hyderabad, India. Asian Federation of Natural Language Processing.

Genitives in Hindi Treebank: An Attempt for Automatic Annotation

Nitesh Surtani, Soma Paul
Language Technologies Research Centre
IIIT Hyderabad
Hyderabad, Andhra Pradesh-500032
nitesh.surtaniug08@students.iiit.ac.in, soma@iiit.ac.in

Abstract

Building syntactic Treebank manually is a time consuming and human labor intensive task. The correctness of annotated data is very important because this resource can be used for developing important NLP tools such as syntactic parsers. In this paper, we examine genitive construction in Hindi Treebank with a view of developing a set of rules for automatic annotation of genitive data in Hindi Treebank. The rules perform quite well producing an overall 89% accuracy for right attachment of genitive noun with its head and correct labeling for the attachment.

1 Introduction

A syntactically annotated Treebank is a highly useful language resource for many NLP tasks including parsing, grammar induction to name a few. Generally, building a Treebank requires an enormous effort by the annotators. But some constructions in Treebank can be automatically annotated. This on one hand reduces the human effort by decreasing the number of intervention required by the annotator, and on other hand helps to maintain consistent annotation. For the automatic annotation of the data, 3 types of cases exist: (1) constructions that have a unique cue that identifies it accurately; (2) construction which occur in varied contexts but still can be identified accurately with well-designed rules; and (3) constructions that cannot be handled using cues. Case 2 constructions are the interesting ones which require special attention for their automatic annotation. Genitive construction in Hindi is one such interesting construction that occurs in varied contexts.

Though, noun with genitive case marker generally modifies a noun, it is also found to occur in other contexts including in relation with verbs, with complex predicates etc. In this paper, we will examine the distribution of genitive data in Hindi dependency Treebank. The aim is to study syntactic cues from the Treebank for determining the legitimate head of the genitive modifier and also identify the relation between the two. We implement the cues as rules for predicting the correct attachment between genitive noun and its head. This is an attempt towards developing semi-automatic annotation of Treebank for the genitive data.

The paper is divided as follows: A detailed study of genitive data in Hindi has been carried out in section 2. Section 3 presents a brief overview of Hindi Treebank and Section 4 talks about distribution of genitives in Hindi Treebank. Section 5 then discusses a rule based approach for automatic annotation of genitive data. The Results and Observation are presented in Section 6 and Section 7 concludes the paper.

2 Genitive Data in Hindi

In Hindi, case markers occur as postpositions. The default genitive case marker specifies a relation between two nouns: head and modifier as in *rAm kA ghar (Ram's house)*, where *rAm* modifies the head *ghar*. The genitive case marker is *kA*, which has allomorphic variations as *kI* and *ke*, governed by the grammatical features of the following head as illustrated in Table 1:

Allomorph	Head Gram. feature	Example
kA	Masculine, Singular, Direct Case	*rAm kA ghar* 'Ram's house'
ke	Masculine, Singular, Oblique Case	*saMwAdAtA ke savAl kA javAb diyA* 'Answered the question of Press'
	Masculine, Plural, Any	*congress kI nIiyAm* 'Policies of Congress'
kI	Feminine, Any	*brahaspativAr kI rAt* 'Thursday's night'

Table 1: Allomorphic variation of the genitive marker in Hindi

As has been widely studied by Girju (2008), the genitive marker between two nouns is highly polysemous in nature as they express different semantic relations. Hindi is no exception in this regard. However, the interesting fact of Indo-Aryan languages and other language families is that genitive data occur in many other contexts. We discuss those contexts in Hindi here one by one. The most significant one is the relation that occurs between the genitive noun and verb as illustrated in (1), which is distinct from (2), which is a regular noun-noun genitive construction.

1. *rAm ke do beTA hE*
   Ram-gen two son be-3sg pr
   'Ram has got two sons.'

2. *rAm ke do beTe skul jA rahe hE*
   Ram-gen two son school go be-3sg pr
   'Two sons of Ram are going to school.'

In (1), the genitive noun is connected to the verb directly and not with the following noun *do beTA* 'two sons'. One might argue for the whole NP *rAm ke do beTe* 'Two sons of Ram' to be argument of the verb *hE* 'is' in the sense of 'There exists two sons of Ram'. This interpretation is not viable because the NP *do beTe* can be scrambled with *hE* as in (3), which is not a regular phenomenon for Noun-gen Noun construction.

3. *rAm ke hE do beTe, aur madhu ke tin*
   Ram-gen be-3sg pr two sons and Madhu-gen three
   'Ram has got two sons and Madhu three.'

However, the case becomes more complex than it was assumed in the beginning because we have come across instances where the head noun is not contiguous with its genitive modifier as exemplified in (4):

4. *mAntriyoM kI samiti kI adyakshaytA rakshA mantrI karenge*
   minister-pl-gen committee-gen chair defense minister do-3sg fut
   'The committee of the ministers will be chaired by Defense Minister.'

A genitive can occur with complex predicate, which is composed of one noun or adjective and a light verb. For example, *pratIkshA karnA* in (5) is a complex predicate because it is a multiword expression denoting single event:

5. *rAm sItA kI pratIkshA kar rahA thA*
   Ram Sita-gen wait do be-3pr pst
   'Ram was waiting for Sita.'

6. *rAm kA jAnA sambhav nahI hE*
   rAm-gen go-VN possible neg be-3pr
   'It is not possible for Ram to go.'

An argument of a verb regularly takes genitive in the context of verbal noun form of a verb. In (6), *rAm* is an argument of the verb *jA* 'go'. The same holds even when some participants intervenes the two as illustrated in (7). Another significant occurrence of genitive is when the head is elided as in (8):

7. *rAm kA sItA ke sAth jAnA sambhav nahI hE*
   rAm-gen Sita with go-VN possible neg be
   'It is not possible for Ram to go with Sita.'

8. *yaha khAnA kal kA hE*
   This food yesterday-gen be-3pr
   'This food is yesterday's (food).'

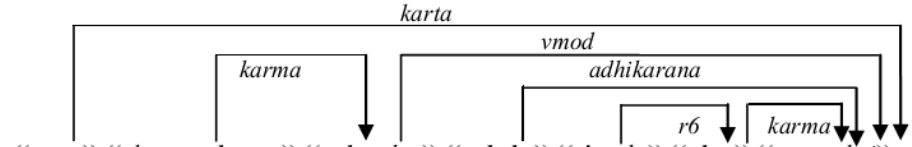
We have examined various distributions of genitive data in Hindi. Table 2 attempts to tabulate all types of genitive that we have discussed in this section:

CASE	CONSTRUCTION TYPE	EXAMPLE
Case 1	Noun gen – Noun	*rAm kA ghar* 'Ram's house'
Case 2	Noun gen – Verb	*rAm kA ek beTA hE* 'Ram has one son'
Case 3	Noun gen – Complex predicate	*rAm sItA kI pratIkshA kar rahA thA* 'Ram was waiting for Sita'
Case 4	Noun gen – Verbal Noun	*rAm kA jAnA* 'Ram leaving'
Case 5	Noun gen – Head elided	*yaha khAnA kal kA hE* 'This (food) is yesterday's food'

Table 2: Different type of genitive data in Hindi

3 Brief Description of Hindi Treebank

The Hindi-Urdu dependency Treebank is being developed following the analysis of the Paninian grammatical model (Bharati et.al. 2009). As observed in Bhatt et al. (2009), "the model offers a syntactico-semantic level of linguistic knowledge with an especially transparent relationship between the syntax and the semantics." The dependency relations are of two types: *kAraka* and non-*kAraka*. *Karaka* relations indicate the roles that various participants play with respect to a verb. There are six *kAraka* relations: *karta*, *karma*, *karana*, *apadana*, *sampradana* and *adhikarana*. *kAraka* relations capture one very significant semantic-pragmatic information which is known as *vivakshA* that can be translated as 'speaker's choice'. Sentences are treated as a series of chunks each having a head and one or more optional modifier of the head as shown below. The head of each chunk is highlighted.



4 Distribution of Genitives in Hindi Treebank

Genitive data is quite frequent in Hindi Treebank. There are total 11505 cases of genitive in a corpus of 10799 sentences (approx 2.5 lakh words). The relation is tagged with a label called r6 that represents the notion of *sashThI sambandha* of Paninian grammar (Bharati et.al 1995). The symbol 'r' indicates that it is not a *kAraka* relation. And the numeral 6 represents *sasThi* (sixth) relation. For all genitives except Case 4 and Case 5 in Table 2, the relation labels contain 'r6' to indicate that the relation is represented by *sasthi* on the surface. For Case 2 in Table 3, the label r6v indicates that the genitive modifier is related to verb and not with any noun as generally is the case with genitives. However, this label is not semantically very informative, which is the case even with the r6 relation. On the other hand, the labels for Case 3, namely r6-k1 and r6-k2, represent both syntactic and syntactico-semantic level of information. When the head noun is derived from verb, the POS tag for such word is given VGNN. The tag implies that the word is noun that has derived from verb. Since, the verbal noun forms retain the verbal property the genitive modifiers of these nouns are tagged with *kAraka* relation. The following table presents distribution of different genitive types in Treebank.

CASE	CONSTRUCTION TYPE	Label	No. of occurrence	%
Case 1	Noun gen – Noun	r6	9123	79.65
Case 2	Noun gen – Verb	r6v	16	0.14
Case 3	Noun gen – Complex predicate	r6_k1	337	2.94
		r6_k2	1595	13.93
Case 4	Noun gen – Verbal Noun	k1	370	3.23
		k2	13	0.11

Table 3: Distribution of genitive data in Hindi Treebank

A genitive marked noun can only take a noun, a verbal noun or a verb as its head. Therefore, all the other POS categories are not the possible candidates for head of a genitive modifier. The default order of genitive modifier and its head is that the modifier precedes the head. Hindi being a free word-order language, we come across cases in the Treebank, where the genitive modifier occurs after the head, which we term here as 'Marked order'. We notice that in the present Treebank, Marked order data is less as shown in Table 4:

Order	Occurrence	%
Default order	11454	99.68
Marked order	37	0.32

Table 4: Distribution of genitive data for Default Order and Marker Order

A genitive noun is contiguous with its head if the position of the head is next to genitive noun. The occurrence of contiguous data in the Hindi Treebank is

quite high. This motivates us towards building a Rule based system for the automatic annotation of the genitive data.

5 A rule based approach for Labeling of Genitive data

Manual development of Treebank is a time consuming and labor intensive task. Attempts have been made by Gupta et.al (2008) to automate some part of the task. A survey of genitive data in Hindi Treebank motivates us to develop set of rules for its automatic annotation. We attempt to predict the correct attachment and the label between the genitive modifier and its head.

5.1 Data Preparation

We have prepared a data set of 633 genitive constructions for testing the rules. Since 'Marked order' data is very less in the Treebank, such data is ignored in the present experiment. The distribution of the test data is kept close to the actual distribution in the Hindi Treebank as shown in Table 5. In the present work, we are not distinguishing different *kAraka* relations for Case 3 and Case 4 because the distinction of *karta, karma* and other *kAraka* relations (if they at all exist) for genitive modifier of complex predicate and verbal noun cannot be syntactically determined. For such distinction, we required deeper semantic knowledge which is beyond the scope of the present paper.

CASE	CONSTRUCTION TYPE	Label	No. of occurrence	%
Case 1	Noun gen – Noun	r6	518	81.8
Case 2	Noun gen – Verb	r6V	6	0.95
Case 3	Noun gen–Complex predicate	r6_k*	92	14.5
Case 4	Noun gen – Verbal Noun	k*	17	2.67

Table 5: Data prepared for testing

5.2 Implementation

The rule based system implements a set of rules for identifying the right attachment and syntactico-semantic label for each attachment. The system matches the rules for the genitive modifier and the candidate chunk and selects the first candidate as the head of the genitive modifier if the rules are matched. The genitive case marker agrees with its head with respect to Gender, Number, Person and Case. The rules verify whether a genitive modifier is followed by a Noun, a verb or a verbal noun or a complex predicate and agrees with its head candidate. Correspondingly, it assigns the label r6, r6v, k* and r6-k*.

6 Result and Observation

We achieve the following result from the rule based system:

CASE	CONSTRUCTION TYPE	Label	Correctly Identified	%
Case 1	Noun gen – Noun	r6	507	97.8
Case 2	Noun gen – Verb	r6V	3	50.00
Case 3	Noun gen – Complex predicate	r6_k*	58	63.04
Case 4	Noun gen – Verbal Noun	k*	11	64.7

Table 6: Results of the rule based approach

The table indicates that the performance for genitive modifier – noun construction is exceptionally good (98%); while for other kind of

construction, we achieve a medium score. This result is encouraging because our Treebank has highest number of representation of this data. If such data can automatically be labeled for correct relation for most of the time, a lot of human labor and time will be saved. The overall result of right attachment and labeling is 89.35%. We note that predicting r6v relation where a genitive noun directly modifies a verb is very difficult because of the non-contiguous occurrence of the genitive modifier and the head in such instance. The main reason for this can be attributed to the greedy selection made by the rule based algorithm, in the sense that it will pick up the first context that one of the rule satisfies without verifying other contexts. For example, given the following sentence, *rAm kA ghar jAnA.* 'Ram's going home', the system will connect *rAm* with *ghar* and assign r6 label without considering the possibility of *ram*'s being connected to *jAnA* which would be the right attachment in this case. To handle this issue, we plan to use a technique that considers all the candidates for head and selects the most probable candidate as the head.

7 Conclusion

The paper presents a detailed study of genitive data in Hindi Treebank. We have examined Hindi dependency Treebank and noted down various syntactico-semantic relations in which genitive noun occurs with respect to its head. We have attempted to trace syntactic contexts which can be used for predicting the relations automatically. The motivation is to automate the process of labeling genitive data. The rule based system implemented in the paper works quite well and predicts the syntactic contexts to a great extent. The output can be verified by the human annotators thus making the Treebank development semi-automatic for the genitive data.

References

[1] A. Bharati, D. M. Sharma, S. Husain, L. Bai, R. Begam and R. Sangal. 2009. *AnnCorra: TreeBanks for Indian Languages, Guidelines for Annotating Hindi TreeBank* (version – 2.0).

[2] R. Girju, (2008) Tutorial on *semantic relation extraction and its applications*. Proceedings of European Summer School in Logic, Language and Information (ESSLLI), Freie und Hansestadt Hamburg, Germany

[3] M. Gupta, V. Yadav, S. Husain and D. M. Sharma. 2008. *A Rule Based Approach for Automatic Annotation of a Hindi Treebank*. 6th International Conference on Natural Language Processing (ICON-08), CDAC Pune, India.

[4] R. Bhatt, B. Narasimhan, M. Palmer, O. Rambow, D. M. Sharma and F. Xia. 2009. *Multi-Representational and Multi-Layered Treebank for Hindi /Urdu*. In Proc. of the Third Linguistic Annotation Workshop at 47th ACL.

[5] Sapna Sharma (2012). *Disambiguating the Parsing of Hindi Recursive Genitive Constructions*. MS Thesis. IIIT Hyderabad, India