Linguistic Issues in Language Technology – LiLT
Submitted, January 2012

Analyzing Parser Errors

To Improve Parsing Accuracy And To Inform Treebanking Decisions

Samar Husain
Bhasha Agrawal

Published by CSLI Publications

LiLT volume 7, issue 11 January 2012

Analyzing Parser Errors

To Improve Parsing Accuracy And To Inform Treebanking Decisions

Samar Husain, *University Of Potsdam, Germany*
Bhasha Agrawal, *Indian Institute Of Information Technology, Design and Manufacturing, Jabalpur, India*

Abstract[1]

We present a detailed error analysis of a transition-based dependency parser trained on a Hindi dependency treebank. Parser error analysis has not been systematically examined from the point of view of treebanking before and this work intends to contribute in this area.

We address two main questions in this paper:

1. Can the parsing of certain structures be made easier by using alternative analyses for these structures?
2. Are there certain linguistic cues implicit (or missing) in the current treebank that can be made explicit (or added) in order to make the parsing of complex constructions easier?

These questions will guide us in examining the potential benefits of parser error analysis during treebanking. Through our experiments and analysis we were able to shed light on the causes of errors and subsequently have been able to improve the performance of the parser.

[1]Substantial amount of this work was completed at Language Technologies Research Centre, IIIT-Hyderabad, India, where the first author was previously located.

LiLT Volume 7, Issue 11, January 2012.
Analyzing Parser Errors.

1 Introduction

Since the availability of Penn Treebank (Marcus et al., 1993), treebanks have played a crucial role in our attempt to build automatic natural language processing tools for various languages. In particular, treebanks have helped in building robust and efficient syntactic parsers. The availability of syntactic parsers is critical for the further processing of a sentence, e.g. semantic analysis (Gildea and Palmer, 2002). The choice of the parsing algorithm or the machine learning strategy can affect parsing accuracy. But this accuracy is also affected by the treebank guidelines. While the analysis of parser errors is used to improve parser performance (by discovering new learning features or re-designing parsing algorithms), it is rarely used to inform guidelines decisions. The common assumption is that treebanks, or more specifically, treebank guidelines capture linguistic realities, while parsers are built to automate these guidelines. But this is strictly not true. We know that many linguistic phenomena have competing (and equally compelling) analyses. Parsers can inform us in judiciously selecting one structure over the other without compromising linguistic expressiveness. In other words, along with using error analysis to improve parser performance, it might be a good idea to incorporate the insights it provides in the treebanking pipeline.

In this paper, we carry out a detailed error analysis of a transition-based dependency parser trained on a Hindi dependency treebank. This analysis is done with respect to 'unlabelled attachment score' (UAS)[2] using four graphical properties, namely, arc type, arc length, arc depth and non-projectivity. The error analysis serves to detect those linguistic patterns that are difficult to parse.

The error analysis helps us formulate the questions that we address in this work.

1. Can the parsing of certain structures be made easier by using alternative analyses for these structures?
2. Are there certain linguistic cues implicit (or missing) in the current treebank that can be made explicit (or added) in order to make the parsing of complex constructions easier?

We are not suggesting that these revisions should be done after the entire treebank is annotated (which would be quite impractical). On the contrary, this exercise can be done on a small subset during the initial treebank development phase. Moreover, one could also investigate whether alternative structures for a linguistic phenomenon can be ob-

[2]Percentage of words that are assigned the correct head

tained from the original analysis deterministically via transformations (we explore this possibility in Section 6).

The obvious benefit of such an exercise is a potential improvement in parser accuracy. But more importantly, this can help the treebank developer in validating various guideline choices by reinforcing decisions that were correct and pointing towards possible revisions. We must note here that while analyzing errors we tried to neglect all the errors caused by algorithmic limitations, learning errors or data sparsity. We understand that doing this is not always trivial; nevertheless, we have tried to focus only on those errors that we thought are due to lack of robust features or to difficult to learn structures. Our work is certainly not without precedent; research in the dependency parsing literature related to feature optimization (Ambati et al., 2010a), (Seeker and Kuhn, 2011), lexicalization (Eryigit et al., 2008), (Kolachina et al., 2011), use of semantics (Bharati et al., 2008), (Ambati et al., 2009), etc. have tried out different types of language specific characteristics and explored ways in which they should be used to influence the parser performance. Similarly, the ideas of pseudo-projective parsing (Nivre and Nilsson, 2005), mildly non-projective structures (Kuhlmann and Nivre, 2006), clausal parsing (Husain et al., 2011) apply some constraints or transform the dependency structures in order to simplify them, thereby making them easier to parse.

In this paper, we briefly describe the Hindi dependency treebank and Hindi dependency parsing in Section 2 and our experimental setup in Section 3. Section 4 shows the classification of parsing errors based on four graphical criteria and this is followed by the error analysis in Section 5. Based on these insights, in Section 6, we explore linguistically competing structures along with potential linguistic cues and alternative ways to encode them. In Section 7 we will see the effect on parsing accuracy. We will discuss the results of our experiments in Section 8.

2 Hindi Dependency Treebank and Parsing

Hindi is a morphologically rich, free word order language (MoR-FWO) with SOV as the default word order. Hindi also has a rich case marking system, although case marking is not obligatory. The Hindi dependency treebank (Begum et al., 2008) is part of a Multi-Representational and Multi-Layered Treebank (Bhatt et al., 2009). The dependency framework (Bharati et al., 1995) used in the dependency treebank is inspired by Panini's grammar of Sanskrit.

Parsing MoR-FWO languages (such as Hindi, Czech, Arabic, Hebrew, etc.) is a challenging task. The difficuly mainly arises because the syntactic cues necessary to identify various relations in these languages are complex. Consequently, parsing accuracies for these languages have not been as high as for fixed word order languages (Nivre et al., 2007a), (Tsarfaty et al., 2010), (Husain et al., 2010). For Hindi, different dependency parsing approaches including both constraint-based (Bharati et al., 2009a), and data driven (Mannem et al., 2009b), (Husain et al., 2010), (Ambati et al., 2010b) ones have been tried. The current state-of-the-art unlabeled attachment score (UAS) hovers around 90% while the labeled attachment score (LAS) is close to 76% for inter-chunk dependency parsing.

3 Experimental setup

All the experiments were conducted using MaltParser[3] (Nivre et al., 2007b). We use the parser settings of Ambati et al. (2010b). We use the dataset[4] released as the part of ICON10 parsing contest (Husain et al., 2010). The training, the development and the test set contained 2972, 543 and 321 sentences respectively. The error classification discussed in the next section is based on the test data. The results reported in Section 7 were obtained using 5-fold cross validation on the complete dataset. The baseline accuracy mentioned in Section 7 differs from the best results of Ambati et al. (2010b) due to differences in the data.

4 Error Classification

The error analysis is split into two parts. In this section, we first classify the errors using 4 criteria. Following this, in section 5, we describe in detail the causes for these errors.

The parser errors for the test data were classified based on the following four criteria:

1. Edge[5] type and Non-projectivity,
2. Edge length,
3. Edge depth.

These properties are known to have a considerable effect on errors in data-driven dependency parsing (McDonald and Nivre, 2007). In this section we quickly summarize some prominent statistics.

[3]MaltParser (version 1.4.1)

[4]We use the inter-chunk dependency trees, rather than expanded trees.

[5]An edge in a dependency tree is the arc that relates two nodes (which are basically words). These are binary asymmetric relations with labels that specify the relation type.

4.1 Edge Type and Non-projective Edge

The 'edge type' class aims at capturing errors based on the particular linguistic phenomenon that they capture. The dependency labels on the edge are indicative of the edge type. Appendix I lists different relations in the treebank that belong to these classes. For example, a noun modifier (nmod) relation is used to represent broad grained noun modification. The linguistic phenomena/concepts were further classified into coarser classes, for example, a 'verb complement' belongs to a class of 'verb argument structure' which in turn belongs to a 'intra-clausal relation' class. The broadest classes that we get using such a classification are *inter-clausal* relations and *intra-clausal* relations. These two classes are based on the notion of clause. We define clause as, *'a group of words containing a single finite verb and its dependents'*. More precisely, let T be the complete dependency tree of a sentence, and let G be a clausal subgraph of T. Then an arc $x \rightarrow y$ in G is a valid arc, if

1. y is not a finite verb;
2. there is no z such that there is an arc $y \rightarrow z$, where z is a finite verb and y is a conjunct.

Other than edge type errors, Table 1 also shows which of these edges are non-projective. Simply put, an arc in a dependency tree is projective if there is a directed path from the head word x to all the words between the two endpoints of the arc (Kübler et al., 2009). The non-projective cases that we found were similar to the ones discussed in Mannem et al. (2009a). A quick look at Table 1 makes few things evident straight away.

1. Most of the errors are clause internal. Such errors amount to ˜83% of all errors.
2. Nearly 50% of all errors are related to verb's argument structure.
3. It is interesting to see that almost half of verb adjunct errors get the correct labels (but of course wrong attachment). This, however, does not hold true for complements.
4. Majority of the non-projective arcs are inter-clausal. The intra-clausal non-projective cases are concentrated in the verb adjunct class.
5. Almost all the relative clause errors are due to non-projectivity.
6. Non-verbal intra-clausal dependency errors make up close to 1/3 of all the errors. Among them, noun modifiers, genitives and coordination are the dominant classes.

Edge Type			No. of edges	No. of non-projective edges	No. of edges with correct label
Main (3.25%)			11	0	0
Intra Clausal (83.13%)	Verb Argument Structure (51.18%)	Complement (21%)	71	4	9
		Adjunct (30.18%)	102	10	45
	Non-verbal (23.67%)	Noun-modifier (14.79%)	50	1	3
		Adjective mod. (1.48%)	5	0	0
		Apposition (0.30%)	1	0	0
		Genitive(7.10%)	24	2	6
	Others (8.28%)	Co-ordination (6.80%)	23	1	0
		Complex Predicate (0.59%)	2	0	0
		Others(0.89%)	3	0	1
Inter Clausal (13.61%)	Co-ordination (0.89%)		3	0	0
	Sub-ordination (12.72%)	Conjunction (1.18%)	4	0	0
		Relative Clause (3.55%)	12	11	0
		Clausal Complement (3.85%)	13	6	0
		Apposition (1.48%)	5	4	0
		Verb Modifier (2.66%)	9	3	0

TABLE 1 Errors based on edge type and non-projectivity.

Length			**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9-20**
Main				1			1	2			7
Intra Clausal	Verb Argument Structure	Complement	10	13	14	6	5	5	3	4	11
		Adjunct	7	23	11	16	8	9	4	8	16
	Non-Verbal	Noun-mod	43	6			1				
		Adjective-mod	5								
		Apposition	1								
		Genitive	12	6	5			1			
	Others	Conjunction	4	13	4	2					
		Complex Predicate	1	1							
		Others	2		1						
Inter Clausal	Co-ordination		1				1				1
	Sub-ordination	Conjunction					1	1	1		1
		Relative Clause	1			2		2	1	2	4
		Clausal Complement	4	4	2				1	1	1
		Apposition	1	3		1					
		Verb Modifier	1			1	1	2	1		3
% of erroneous arcs*			6	16	13	12	10	15	13	18	24

TABLE 2 Errors based on edge length. (* rounded off)

4.2 Edge Length

Edge length corresponds to the linear distance between a head and its child. Table 2 classifies the errors based on this distance. For each arc length, the table also shows the percentage of wrong arcs. The following observations are based on the data in Table 2:

1. In general, as the length increases the percentage of errors increases. Hence, while only 6% of all arcs with length = 1 are incorrect, 24% of all arcs with length = 9 to 20 are incorrect.
2. More than 50% of verbal complement errors are caused by 1 to 4 edge lengths. (Although, the error rate for adjuncts is evenly distributed across lengths 1 to 20).
3. In spite of having adjacent elements, noun modification and genitive relations are not being correctly identified.
4. The majority of inter-clausal verb modifier errors are long distance.

4.3 Edge Depth

The depth of a tree is the total number of levels of links that it has. Consequently, the depth of an edge is the level at which it is situated in the tree. Due to space constraints we cannot show the complete statistics. But here are the main observations:

1. Most of the intra-clausal dependency errors appear at depths 1 to 6.
2. Inter-clausal errors are mostly at depth 1 and 2.

5 Error Analysis

In this section we will discuss the causes for the different parser errors. As mentioned earlier, during our analysis we have tried to neglect all the errors caused by learning or data sparsity. We have tried to focus only on those errors that are due to a lack of robust features or to difficult to learn structures. Below, we have listed the causes of both intra-clausal and inter-clausal errors (** signifies high occurrence).

5.1 Intra-clausal errors

Verbal complements and adjuncts

Most of the intra-clausal verbal errors are due to the following reasons:

- A coordinating conjunction (e.g. *Ora* 'and') becomes a child (either as a complement or an adjunct) of a verb. For example, in sentence (1)[6], identifying *Ora* as the child of *deKI* 'saw' is not easy.

  (1) aBaya Ora aBiSeka ne pikCara deKI
  Abhay and Abhishek ERG movie saw
  'Abhay and Abhishek saw a movie.'

- ** Attachment ambiguities for a child due to intervening non-finite verbs. For example, in (2) below, *kala* 'yesterday' can be attached to either the non-finite verb (*soCawe hue* 'while thinking') or the main verb (*deKI* 'saw'). Frequently, this also leads to long distance arcs.

  (2) kala aBaya Ora aBiSeka ne soCawe hue pikCara
  Yesterday Abhay and Abhishek ERG thinking be-prog movie
  deKI
  saw
  'Abhay and Abhishek saw a movie yesterday while thinking'

- ** Ambiguous post-position of a nominal child node. For example, the post-position *se* can act as 'INSTRUMENT', 'DATIVE', or 'CAUSE' depending on the context. Similarly, *para* is ambiguous between 'PLACE', and 'TIME'. Hindi has plenty of such ambiguous post-positions

[6] In the notation used for transcribing Hindi, capitalization roughly represents aspiration for consonants and longer length for vowels. In addition, 'w' represents 't' as in French *entre* and 'x' something similar to 'd' in French *de*.

- ** Lack of post-position on the nominal child. This is similar to the above point, in a sense that this also leads to ambiguity.
- ** The non-finite clause gets split due to shared arguments. In (3) for instance, the non-finite clause consists of *xeKakara* 'after seeing' and its argument *pikcara* 'movie' but *aBaya* and *kala*, which are dependents of the main verb *soyA*, are found between this argument and the verb.

(3) pikcara aBaya kala xeKakara soyA
movie Abhay yesterday after-seeing slept
'Abhay slept yesterday after finishing the movie'

- Embedded finite clause (apposition) creates a long distance dependency between the child and its head, and thereby a wrong attachment.
- The nominal child appears in the non-prototypical direction (to the right, instead of left) with respect to its head verb. In (4), for example, the subject of *soyA* 'slept', *aBaya* appears on the right of the verb instead of left (cf. Example (3)).

(4) kala soyA aBaya
Yesterday slept Abhay
'Abhay slept yesterday'

Noun modifications (including apposition and genitive)

Most of the errors in nominal modifications were due to the following reasons. (The statements preceded by GEN are specific to the genitive relation between two nominals.)

- ** Nominal modification errors are due to the absence of a post-position on the child. This occurs with names, appositions, etc.
- When a noun becomes the head of a gerund or an adverb, identifying the noun as the head is difficult due to lack of robust cues.
- ** GEN: Coordination creates problem in nominal modifications as well. This is true for coordinating conjunctions becoming the head or a child in a genitive relation. In (5), for example, the coordinating conjunction *Ora* 'and' should be the child of *Gara* 'home', but this is not done correctly.

(5) aBaya Ora rakaSanxA kA Gara acCA hE
Abhay and Rakshanda GEN home good is
'Abhay and Rakshanda's house is good.'

- ** GEN: Some modifiers intervene between the child and the head. Notice how in (6) *xilli meM sWiwa* 'situated in Delhi' intervenes between *Ora* and *Gara*.

(6) aBaya Ora rakaSanxA kA xilli meM sWiwa Gara acCA
Abhay and Rakshanda GEN Delhi IN situated home good
hE
is
'Abhay and Rakshanda's Delhi house is good.'

- GEN: Splitting of the genitive post-position and the head noun or scrambling of the head noun generally lead to errors.

Coordination

As we have observed above identifying the correct incoming arc of a conjunct is generally quite difficult. And this is not surprising as, among other things, the conjunct itself does not have any explicit features (gender-number-person features or post-position information). Here are some more causes:

- ** The number of children of a coordinating conjunction is not fixed (it can be more than 2). Consequently these children can be spread across the entire sentence (leading to long distance dependencies). Moreover, there are no robust cues to identify these children (e.g. commas are not always present).
- ** Children can be sub-trees (e.g. a coordination of multiple non-finite verbs that take their own arguments, genitives, etc.). This again creates long distance dependencies.
- Coordination can also involve attachment ambiguities (e.g. genitives).
- Coordination can sometimes lead to non-projectivity.

Complex predicates

Complex predicates are generally identified correctly if the noun-verb or adjective-verb occur together. In other cases, they are difficult to identify (Begum et al., 2011).

5.2 Inter-clausal errors

The most common source of error in inter-clausal relations is non-projectivity. Complex interactions of different conjunctions that can lead to long distance dependencies are another source of error. Clause internal scrambling of subordinating conjunctions can also lead to error.

6 Experiments

In Section 4 and Section 5, we classified and describeed the errors. We also discussed their possible causes in Section 5. Errors related to the verb's argument structure (cf. Table 1) were amongst the most frequent error types. We noted that the absence of post-positions, ambiguous

	Transformation	Transformation Cue Availability	Inverse Transformation	Use of a tool
1	Paired Connective	Yes	Yes	No
2	Relative Clause	Yes	Not 100%	MaxEnt
3	Complement Clause	Yes	Yes	No
4	Intra-clausal Coordination	Yes	Yes	No
5	Complex Predicate argument (with genitive case marker)	Yes	Yes	No

TABLE 3 Experiment I: Alternative structures using structure transformation.

post-positions, coordination, etc are some of the reasons for these errors. The other major source of errors was non-projectivity. This was the main cause for errors in relative clause constructions and sentences with clausal complements(cf. Table 1). As discussed in Section 1, the observations from error analysis are used in two ways,

1. to simplify dependency structures,
2. to incorporate additional linguistic information and investigate ways to represent/encode this information.

The first set of experiments target non-projective dependency trees (along with some projective ones). We explore the possibility of making these structures projective and also whether the original structure can be successfully recovered after parsing (this is done using pseudo-projective parsing (Nivre and Nilsson, 2005)). If the projective counterpart is linguistically sound, one could argue that the guideline decisions for such phenomenon can be revised. The second set of experiments either explores alternative ways to encode the already available treebank information or it tries to use additional knowledge to add more linguistic information in the treebank. This set of experiments highlights different encoding strategies and points to the need for additional linguistic information to improve parsing accuracy.

6.1 Experiment I: Exploring alternative analysis via structural transformations

Transformations lead to structural changes in a dependency tree when relations (and possibly relation types) between nodes are modified leading to a new dependency tree. Table 3 summarizes all the transformations. The first three transformations remove non-projectivity in paired connective, relative and complement clauses. These three non-projective cases are also the most frequent non-projective structures in Hindi (Mannem et al., 2009a). Figure 1 shows the changes in the analysis of paired connectives[7]. A possible transformation for relative clause structures is shown in Figure 2. The inverse transformation for relative clause structures is not 100% accurate; we use a MaxEnt[8] based boolean classifier to automatically identify the head of the relative clause and lexical cues to identify the relative clauses themselves. This is similar to the method used in Husain et al. (2011). The 3^{rd} transformation in table 3 removes non-projectivity in sentences such as (7) where the non-finite verb (*kehanA* 'saying') takes a clausal complement. The arc between this verb and the verb of the clausal complement is non-projective. The transformation establishes an arc between the main verb (*hE* 'is') and the clausal complement instead.

(7) aBaya kA kehanA hE ki kala bAriSa hogI
Abhay GEN saying is that tomorrow rain happen
'According to Abhay, it will rain tomorrow.'

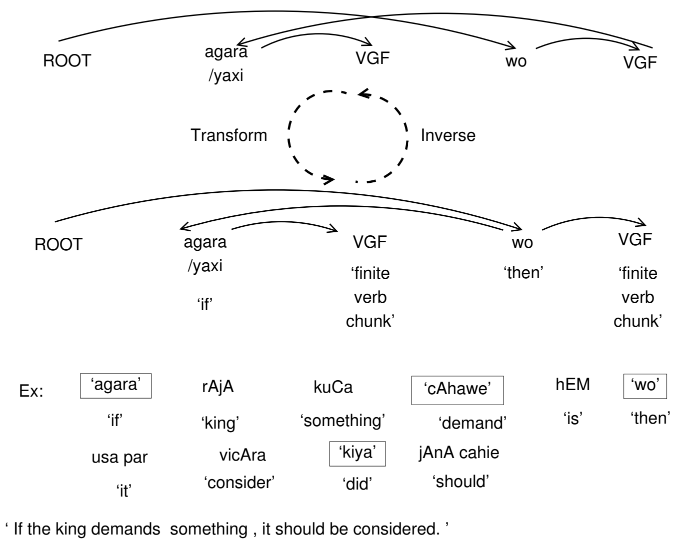

FIGURE 1 Alternative analysis for a sentence with paired connectives via structural transformation

Using the 4^{th} transformation we try an alternative structure for intra-clausal coordination. The current analysis treats the conjunction as the head and all the nominals as its children. We experiment with an

[7]Discussions during the Hindi-Urdu treebank meeting at Boulder in July 2011 were instrumental in formulating the alternative analysis of paired connectives.
[8]http://maxent.sourceforge.net/

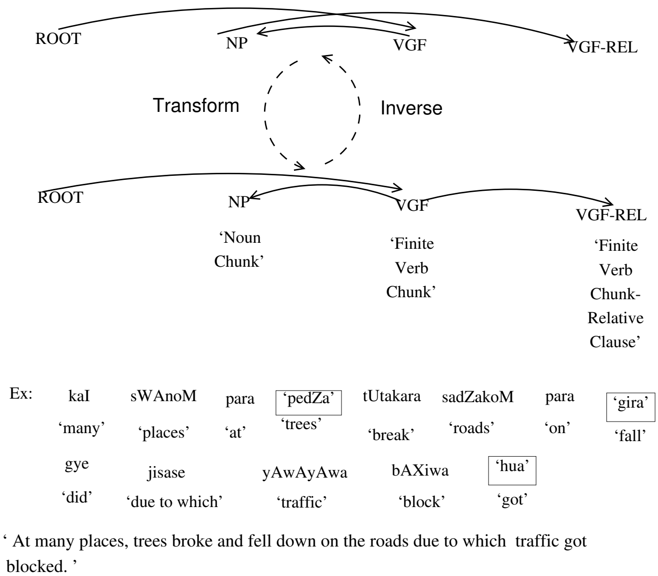

FIGURE 2 Alternative analysis for a sentence with a relative clause via structural transformation

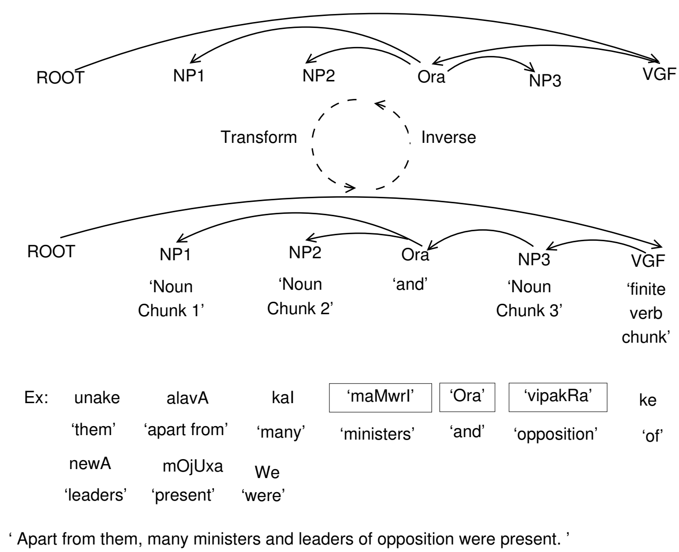

FIGURE 3 Alternative analysis for a sentence with intra-clausal coordination via structural transformation

	Addition/ Modification	Encoded through	Cue Availability	Inverse	Use of a resource
1	Encoding verb-argument agreement	Dependency label	Yes(morph)	Yes	No
2	Encoding conjunction valency	POS Tag	Yes (lexical)	Yes	Valency lexicon
3	Encoding verbal valency	POS Tag	Yes (lexical)	Yes	Bilingual Dictioanry, VerbNet

TABLE 4 Experiment II: Encoding linguistic information

analysis where the rightmost child of a conjunct is treated as the head, the conjunction now becomes its child. The children on the left are still attached to the conjunct. This analysis is shown in Figure 3. The motivation is to make the post-position of the rightmost child explicit during parsing (cf. co-ordination in Section 5.1). The 5^{th} transformation looks at the attachment of arguments (with genitive post-positions) of complex predicates. Currently, they are attached to the nominal predicate. We experiment with attaching them directly to the light verb. The motivation here is to reduce the confusion with the genitive relations which have the same post-position (cf. verbal complements in Section 5.1).

The alternative analyses for paired connectives, intra-clausal coordination, and complex predicate argument are as compelling as the present analyses for these structures. Interestingly all the alternative structures (except for relative clause sentences) can be obtained from one another deterministically.

6.2 Experiment II: Encoding linguistic information

Analysis of a sentence requires the identification of various linguistic information present in a sentence. Such linguistic information can correspond to cues that are lexical, morphological, morpho-syntactic, syntactic, etc. But does the treebank provide all the cues necessary for analysis? Analyzing a sentence not only involves identification of such information but also exploring ways to encode this information during annotation. If there are multiple ways to encode some information, is one encoding strategy better than the other?

Table 4 summarizes this set of experiments. The 1^{st} subset here tries to make the agreement features visible via a dependency label. Previous work on Hindi dependency parsing (Bharati et al., 2008), (Ambati et al., 2010b) has found that the gender, number, person (gnp) feature crucial for agreement remains unexploited during training. In section 5.1 we mentioned that many errors are due to the lack of post-positions. As agreement in Hindi for both subject and object is blocked by post-positions, we aim, through this experiment, to reduce some errors caused by lack of post-positions. Note again that the gnp feature is already available in the treebank, in this experiment we make use of this feature to encode agreement differently. This is done via a change in the dependency label of the argument with which the verb agrees. Take Example 8 for instance,

(8) aBaya ne KIra KAI
Abhay ERG sweets ate
'Abhay ate sweets'

In this example, 'Abhay' is the subject (dependency label 'k1') and 'sweets' is the object (dependency label 'k2'). The main verb ('ate') agrees with the object, so the dependency label of 'sweets' is modified from 'k2' to 'k2_agr' to capture agreement. Similarly, when the verb agrees with the subject, dependency label of subject is modified from 'k1' to 'k1_agr'. In this way, dependency relations can be used to capture agreement.

In the 2^{nd} subset we try to make a distinction between the POS tags for coordinating and subordinating conjunction. This is done by using a lexical resource. Currently coordinating and subordinating conjunctions have the same CC tag. But the distinction is in fact made in a constraint-based parser for Hindi (Bharati et al., 2009a) and we want to see if this will help data driven parsing. Finally, the 3^{rd} subset incorporates verb valency information in its POS tag. Currently only a single tag is used for all the verbs. We introduce VM-1, VM-2 and VM-3 for intransitive, transitive and ditransitive verbs respectively. This is done using an English-Hindi bilingual dictionary[9] and VerbNet (Kipper et al., 2006). Through this information, we hope to reduce the argument structure errors shown in Table 1.

7 Effect of Experiments I and II on parsing accuracy

Table 5 shows the parser accuracy in terms of UAS, LAS and LA[10] for all the experiments. The experiments that gave us statistically significant improvement have been marked with an asterisk (*).

Table 5 shows that only structural changes led to improvements in the accuracy. Other experiments that made changes in dependency labels, POS tags, etc. to encode valency information and agreement did not lead to any improvements. But we think that the low coverage (only 196 verbs out of 267 were found in the dictionary) of the bilingual dictionary for verbs affected the experiment where we tried to

[9] http://ltrc.iiit.ac.in/onlineServices/Dictionaries/Dict_Frame.html
[10] LAS/UAS/LS = Percentage of words assigned correct head+label/head/label

	Experiments	LAS	UAS	LA	Statistical Significance
	Baseline Accuracy	77.58	88.97	80.48	-
Experiment I	*Paired Connectives	77.70	89.15	80.61	UAS,LAS,LA
	*Corelative and extraposed relative clauses	77.59	89.02	80.72	LA
	Clausal Complement	77.47	88.89	80.39	-
	Complex predicate argument (with genitive case marker)	77.50	88.77	80.43	-
	*Intra Clausal coordination	78.21	89.06	81.21	LAS,LA
Experiment II	Encoding agreement via dependency labels	74.15	88.87	79.98	-
	Encoding conjunction valency through POS	77.60	89.00	80.46	-
	Encoding verb's valency through POS	77.53	88.92	80.42	-

TABLE 5 Effect of Experiment I and II on parser accuracy. * shows statistical significance with McNemar's test ($p <= 0.01$), computed using MaltEval (Nilsson and Nivre (2008)).
UAS: Unlabeled Attachment Score, LAS: Labeled Attachment Score, LA: Label Accuracy

encode verbal valency. Encoding conjunction valency also did not lead to any improvement. The results show that the lexical information for conjunctions in itself is sufficient to disambiguate the coordination vs. subordination structures correctly and the added valency information seems to be redundant.

8 Discussion

We carried out a detailed error analysis and attempted two sets of experiments, the first that involved structural simplification via transformations and the second that involved linguistic encoding through which we showed that linguistic generalizations (eg. agreement) can be encoded during the annotation task in more than one way. The second set also investigated the need for additional linguistic information to improve parsing accuracy. The result showed that taking into account competing linguistic analyses is primarily responsible for the improvement of parser accuracy. Given these observations it is easy to see that analysis of parser errors and performing experiments such as those mentioned in section 6 can play an important role in the evolution of treebanking guidelines decisions. One could argue that this will eventually make treebanks biased to specific parser developments. But as long as the linguistic integrity of the analysis is maintained, this will not be a disadvantage- after all, most treebanks are overwhelmingly used to develop high performance automatic parsers. For the Telugu dependency treebank Kolachina et al. (2011) manually annotated the phenomenon of external 'sandhi'. A transition-based dependency parser trained on this modified treebank performed better than the one trained on the old version. But more importantly, their decision to incorporate 'sandhi' information in the treebank was a result of a detailed error analysis of parser (trained on the older version treebank). Similarly, parser error analysis led to the annotation of some semantic tags in a Hindi dependency treebank (Bharati et al., 2008), (Ambati et al., 2009). The use of these semantic tags then led to an improvement in parser accuracy as it was possible to untangle the confusion caused by lack of nominal post-positions. Some of these semantic tags were helpful in precisely those cases where surface cues were unavailable. More recently, Seeker and Kuhn (2011) investigated the role and ways of incorporating morpho-syntactic information in a German treebank and its effect on parsing accuracy. They found that while lexicalization is sufficient to recover the internal structure of noun phrases, explicit morpho-syntactic features help in better selection of grammatical functions. This research clearly shows that analyzing parsing errors does help to determine better ways of representing/encoding linguistic features in a treebank or in deciding what features are required for better parsing. This not only leads to an increase in parser accuracy but also helps create better treebanks.

Acknowledgments

Thanks to the three anonymous reviewers for their detailed reviews. We would also like to thank Ashwini Vaidya and Annie Zaenen for comments that helped us improve the quality of the paper.

References

Ambati, B., S. Husain, S. Jain, Sharma D., and R. Sangal. 2010a. Two methods to incorporate 'local morphosyntactic' features in hindi dependency parsing. In *NAACL HLT 2010 workshop on SPMRL, Los Angeles, CA*.

Ambati, B.R., S. Husain, J. Nivre, and R. Sangal. 2010b. On the role of morphosyntactic features in hindi dependency parsing. In *NAACL HLT 2010 workshop on SPMRL, Los Angeles, CA*.

Ambati, B. R., P. Gade, Chaitanya GSK, and S Husain. 2009. Effect of minimal semantics on dependency parsing. In *RANLP09 student paper workshop*.

Begum, R., S. Husain, A. Dhwaj, S. Sharma, and R. Bai, L.and Sangal. 2008. Dependency annotation scheme for indian languages. In *IJCNLP08*.

Begum, R., K. Jindal, A. Jain, S. Husain, and S. Sharma. 2011. Identification of conjunct verbs in hindi and its effect on parsing accuracy. In *In Proceedings of the 12th CICLing*.

Bharati, A., V. Chaitanya, and R. Sangal. 1995. Natural language processing: A paninian perspective. In *Prentice-Hall of India, New Delhi, pp. 65-106*.

Bharati, A., S. Husain, B. Ambati, S. Jain, D.M. Sharma, and Sangal. R. 2008. Two semantic features make all the difference in parsing accuracy. In *Proceedings of the 6th ICON*.

Bharati, A., S. Husain, D. Misra, and R. Sangal. 2009a. Two stage constraint based hybrid approach to free word order language dependency parsing. In *11th IWPT09. Paris*.

Bharati, A., D. M. Sharma, S. Husain, R. Bai, R. Begum, and S. Sangal. 2009b. Treebanks for indian languages, guidelines for annotating hindi treebank (version 2.0). http://ltrc.iiit.ac.in/MachineTrans/research/tb/DS-guidelines/DS-guidelines-ver2-28-05-09.pdf.

Bhatt, R., B. Narasimhan, M. Palmer, O. Rambow, D. Sharma, and F Xia. 2009. A multi-representational and multi-layered treebank for hindi/urdu. In *Proceedings of 3rd LAW (ACL-IJCNLP 2009)*.

Eryigit, G., J. Nivre, and K. Oflazer. 2008. Dependency parsing of turkish. In *Computational Linguistics 34(3), 357-389*.

Gildea, D. and M. Palmer. 2002. The necessity of parsing for predicate argument recognition. In *Proceedings of the 40thMeeting of the Association for Computational Linguistics*.

Husain, S., P. Gadde, J. Nivre, and R. Sangal. 2011. Clausal parsing helps data-driven dependency parsing: Experiments with hindi. In *In Proceedings of IJCNLP 2011*.

Husain, S., P. Mannem, B. Ambati, and P. Gadde. 2010. The icon-2010 tools contest on indian language dependency parsing. In *ICON10 NLP Tools Contest on Indian Language Dependency Parsing*.

Kipper, K., A. Korhonen, N. Ryant, and M. Palmer. 2006. Extensive classifications of english verbs. In *Proceedings of the 12th EURALEX International Congress*.

Kolachina, S., D.M. Sharma, P. Gadde, M. Vijay, R. Sangal, and A. Bharati. 2011. External sandhi and its relevance to syntactic treebanking. In *Proceedings of CICLing 2011, Tokyo, Japan*.

Kübler, Sandra, Ryan T. McDonald, and Joakim Nivre. 2009. *Dependency Parsing*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Kuhlmann, M and J Nivre. 2006. Mildly non-projective dependency structures. In *Proceedings of the COLING/ACL*.

Mannem, P., A. Abhilash, and A. Bharati. 2009b. Ltagspinal treebank and parser for hindi. In *Proceeding of ICON09*.

Mannem, P., H. Chaudhry, and A. Bharati. 2009a. Insights into non-projectivity in hindi. In *ACL-IJCNLP Student Research Workshop*.

Marcus, M., B. Santorini, and M.A. Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. In *Computational Linguistics 1993*.

McDonald, R. and J. Nivre. 2007. Characterizing the errors of data-driven dependency parsing models. In *EMNLP/CONLL*.

Nilsson, J. and J. Nivre. 2008. Malteval: An evaluation and visualization tool for dependency parsing. In *the Sixth LREC, Marrakech, Morocco*.

Nivre, J., J. Hall, S. Kubler, R. McDonald, J. Nilsson, S. Riedel, and D. Yuret. 2007a. The conll 2007 shared task on dependency parsing. In *Proc of EMNLP/CoNLL-2007*.

Nivre, J., J. Hall, J. Nilsson, A. Chanev, G. Eryigit, S. Kubler, S. Marinov, and E. Marsi. 2007b. Maltparser: A language-independent system for data-driven dependency parsing. In *Natural Language Engineering*.

Nivre, J. and J. Nilsson. 2005. Pseudo-projective dependency parsing. In *Proc. Of ACL 2005*.

Seeker, W. and J. Kuhn. 2011. On the role of explicit morphological feature representation in syntactic dependency parsing for german. In *Proceedings of IWPT 2011*.

Tsarfaty, R., D. Seddah, Y. Goldberg, S. Kuebler, Y. Versley, M. Candito, Foster J., I. Rehbein, and L. Tounsi. 2010. What, how and wither. In *NAACLHLT 2010 workshop on SPMRL.*.

Appendix

Edge Type			Dependency Label
Main			main
Intra Clausal	Verb Argument Structure	Complement	k1, pk1, jk1, k1s, k1u, k2, k2p, k2g, k2u, k4, k4a, k4u, r6-k1, r6-k2
		Adjunct	k2s, k3, k5, k7, k7p, k7t, vmod, vmod__adv, rh, rt, adv, ras-k1, rd, ras-NEG, ras-k2, sent-adv, k7a
	Non-Verbal	Noun-mod	nmod, nmod_k1inv, nmod_k2inv, nmod_pofinv
		Adjective-mod	jjmod
		Apposition	rs
		Genitive	r6, r6v
	Others	Conjunction	ccof
		Complex Predicate	pof, pof_idiom
		Others	fragof
Inter Clausal	Co-ordination		ccof
	Sub-ordination	Conjunction	ccof
		Relative Clause	nmod__relc
		Clausal Complement	k2, k2s, k1s
		Apposition	rs
		Verb Modifier	vmod, vmod_adv, sent_adv, rh

TABLE 1 Dependency relations and corresponding multi-level classes. For details on the dependency labels and the annotation framework see (Bharati et al., 2009b)