Polibits

ISSN: 1870-9044

polibits@nlp.cic.ipn.mx

Instituto Politécnico Nacional

México

Kolachina, Sudheer; Misra Sharma, Dipti; Gadde, Phani; Vijay, Meher; Sangal, Rajeev; Bharati, Akshar

External Sandhi and its Relevance to Syntactic Treebanking

Polibits, vol. 43, 2011

Instituto Politécnico Nacional

Distrito Federal, México

External Sandhi and its Relevance to Syntactic Treebanking

Sudheer Kolachina, Dipti Misra Sharma, Phani Gadde, Meher Vijay,
Rajeev Sangal, and Akshar Bharati

***Abstract*—*External sandhi* is a linguistic phenomenon which refers to a set of sound changes that occur at word boundaries. These changes are similar to phonological processes such as assimilation and fusion when they apply at the level of prosody, such as in connected speech. External sandhi formation can be orthographically reflected in some languages. External sandhi formation in such languages, causes the occurrence of forms which are morphologically unanalyzable, thus posing a problem for all kind of NLP applications. In this paper, we discuss the implications that this phenomenon has for the syntactic annotation of sentences in Telugu, an Indian language with agglutinative morphology. We describe in detail, how external sandhi formation in Telugu, if not handled prior to dependency annotation, leads either to loss or misrepresentation of syntactic information in the treebank. This phenomenon, we argue, necessitates the introduction of a sandhi splitting stage in the generic annotation pipeline currently being followed for the treebanking of Indian languages. We identify one type of external sandhi widely occurring in the previous version of the Telugu treebank (version 0.2) and manually split all its instances leading to the development of a new version 0.5. We also conduct an experiment with a statistical parser to empirically verify the usefulness of the changes made to the treebank. Comparing the parsing accuracies obtained on versions 0.2 and 0.5 of the treebank, we observe that splitting even just one type of external sandhi leads to an increase in the overall parsing accuracies.**

***Index Terms*—Syntactic treebanks, sandhi.**

I. Introduction

In recent years, there has been a steady increase in awareness about the multi-fold importance of treebank corpora. This is evident from the number of syntactic treebanking projects for different languages that have been initiated and are currently ongoing. Although initial efforts such as the Penn treebank (PTB) [1] worked with constituency-based representations, many treebanking efforts in the last decade have preferred dependency-based representations. Two main reasons for this preference can be gathered from the literature. The first reason is that for languages that are relatively free word-order, dependency-based representation has been observed to work better [2], [3], [4], [5]. The second is that the simplicity of dependency-based representation makes it amenable for a variety of natural language processing applications [6], [7].

A dependency annotation scheme inspired by Paṇinian theory was proposed for syntactic treebanking of Indian languages (ILs) which are both morphologically rich and relatively free word-order [8]. This scheme has hitherto been applied to three Indian languages: Hindi, Bangla and Telugu. The first versions of all three treebanks were released for the shared task on IL parsing held as part of ICON-2009[1]. While the Hindi treebanking effort has matured considerably and the treebank is being developed at a stable pace [9], [10], [11], the Telugu and Bangla treebanks are still at a very initial stage of development. In this paper, we discuss some issues in Telugu treebanking with special reference to a linguistic phenomenon known as *external sandhi*. We discuss how external sandhi formation in Telugu poses problems during syntactic annotation of Telugu sentences. We discuss how this language-specific issue necessitates the introduction of a sandhi splitting or segmentation stage in the generic annotation pipeline. As a preliminary step towards a sandhi split treebank of Telugu, we manually split all instances of one type of external sandhi widely occurring in the previous version of the treebank. We conduct an experiment to empirically verify the efficacy of sandhi splitting in the Telugu treebank for the task of NL parsing.

II. Related Work and Background

Begum et al. [8] proposed a dependency-based annotation scheme for syntactic treebanking of Indian languages. Currently, treebanks of three Indian languages[2], Hindi, Bangla and Telugu are being developed using this annotation scheme. All these three languages are morphologically rich and have a relatively free word order. The applicability of this scheme for syntactic annotation of a fixed word order language like English has also been studied to some extent [12], [13]. The annotation scheme is based on a grammatical formalism known as Computational Paṇinian Grammar (CPG), a brief introduction to which is given in the next section. Vempaty et al. [14] give a detailed account of the issues encountered in the application of this annotation scheme to the treebanking of Telugu along with the decisions taken to address each of

Manuscript received October 27, 2010. Manuscript accepted for publication January 14, 2011.

The authors are with the Language Technologies Research Centre, IIIT-Hyderabad, India (e-mail: {sudheer.kpg08, phani.gadde, mehervijay.yeleti}@research.iiit.ac.in, {dipti, sangal}@mail.iiit.ac.in)

[1]NLP Tools Contest on IL parsing. *http://ltrc.iiit.ac.in/nlptools2009/*

[2]HyDT-Hindi, HyDT-Bangla and HyDT-Telugu

them in the development of version 0.1 of the treebank. They also discuss a few syntactic constructions in Telugu which are of interest from the parsing perspective.

Telugu is a Dravidian language with agglutinative morphology [15]. Although all Indian languages in general are said to be morphologically rich and therefore have relatively free word order, there exist considerable differences among them with respect to their finer morphological properties. For instance, the morphology of Hindi, an Indo-Aryan language is said to be inflectional as one morph can be mapped to several morphemes. However, inflectional morphs in Hindi such as case-markers and auxiliaries are not bound to their stems. This is typically considered a property of languages with analytical morphology. Telugu, on the other hand, is characterized as having an agglutinative morphology. It must be noted that agglutination in its original formulation [16], refers to the property of a one-to-one mapping between morphs and morphemes. In Telugu, inflectional morphs (which include different kinds of auxiliary verbs and case-markers) are always bound to the stem resulting in highly synthetic word forms. The number of possible verb forms for a verb stem in Telugu therefore, is very high, aggravating the task of the morph analyzer. In addition, as we will show through example sentences in section IV, even full morphological words can fuse together in Telugu resulting in complex forms which are morphologically unanalyzable[3]. Such complexities in the morphology of Indian languages point towards the need for a more exact approach while typifying them similar perhaps, to the one espoused in Greenberg [17].

The notion of external sandhi in traditional Sanskrit grammars captures well the phenomenon of word fusion mentioned above. In section IV, we show how this phenomenon, if not addressed through sandhi-splitting, poses problems for syntactic analysis of Telugu sentences during treebanking. This phenomenon was not handled in the preliminary version of the Telugu treebank[4] released for the shared task on IL parsing at ICON 2009. Although the number of sentences in the Telugu treebank (1400 sentences for training, 150 sentences each for development and testing) was comparable to that of the Hindi and the Bangla treebanks[5], the average parsing accuracy for Telugu on both coarse and fine-grained datasets was much lower as compared to the other languages [18]. In fact, all the participating systems reported their lowest accuracies on the Telugu datasets. An analysis of the Telugu treebank was carried out in order to discover possible reasons for such low accuracies on the parsing task. As a result, two possible reasons were identified. One reason for the relatively low accuracies on the Telugu datasets was that there was a considerable difference of domain between the training set on the one hand, and the development and test sets on the other. Such ill-effects of domain differences on the parsing accuracies can be easily avoided by partitioning the treebank differently and are hence, not a source of worry during treebank development.

TABLE I
IL TREEBANK STATISTICS: A COMPARISON

Language	sentences	words / sentence	chunks / sentence
Hindi	1800	19.01	9.18
Bangla	1280	10.52	6.5
Telugu	1700	5.43	3.78

The second reason was the lower number of both words and chunks in Telugu sentences as compared to Hindi and Bangla (shown in Table I[6]). The shared task dealt with chunk parsing which means that dependencies are shown only among the chunks[19] in a sentence. In the case of Telugu, as the above table shows, different syntactic relations are possible with the same dependency structure leading to sparsity. Statistical parsers have to learn the same number of syntactic relations from relatively lesser structure in Telugu as compared to Hindi and Bangla. We show in section IV that the smaller number of chunks per sentence in Telugu is directly attributable to the phenomenon of external sandhi formation.

It must be mentioned that in terms of parsing accuracies, a similar pattern was observed in the case of Turkish at both the CONLL shared tasks on dependency parsing [20], [21]. Interestingly, Turkish is also an agglutinating language with relatively free word order. In fact, the agglutinative morphology of Turkish dictated the choice of the treebank architecture used in the development of the Turkish treebank [5]. In the Turkish treebank, the complex agglutinative word forms are represented as sequences of *inflectional groups* separated at derivational boundaries. Syntactic relations are represented between these inflectional groups rather than between word forms. This information about word structure is preserved because morphological features of intermediate derivations are cues for syntactic relationships. Thus, annotation of syntactic dependencies is done on top of a tier of morphological segmentation. It would be interesting to do a detailed comparison of the annotation schemes of the Turkish and Indian language treebanking efforts vis-a-vis the properties of these languages.

In another related work, external sandhi has been recently discussed in the context of building an automatic Sanskrit segmentizer [22]. In this work, two different approaches to automatic segmentation are explored, both of which perform with reasonable accuracy.

III. CPG FORMALISM

The CPG formalism is a dependency grammar inspired, as mentioned previously, by Paṇinian grammatical theory. In this

[3]It would not be correct to call instances of such fusion as 'word forms' as they fall outside the domain of morphology. We argue in section IV that they are a result of orthographic expression of prosodic processes.

[4]version 0.2

[5]Hindi: 1500 training, 150 development, 150 testing
Bangla: 980 training, 150 development, 150 testing

[6]Note that the comparison shown here is based on the datasets released for the shared task on parsing held at ICON 2009.

formalism, as in other dependency grammars, the syntactic structure of a sentence in a natural language consists of a set of binary asymmetric relations called *dependencies* between the 'words' (lexical items) in that sentence. A dependency relation is always defined between a *head* word and a *modifier* word that modifies the head. In the CPG formalism, the verb is treated as the head of a clause. Nouns denoting the participants in the activity denoted by the verb stem are treated as modifiers of the verb. The relation between a verb and its modifier is known as a *karaka* relation, a notion central to syntactic analysis in Paṇinian grammar. *Karaka* relations are syntactico-semantic relations that obtain between a verb and its modifier. Each participant of the activity denoted by a verbal stem is assigned a distinct *karaka*. For example, $k1$ or *karta* is a relation that picks out the participant most central to the activity denoted by the verb. There are six different *karaka* relations defined in Paṇinian grammar. In addition to *karaka* relations that obtain between verbs and their participants, dependency relations can also exist between pairs of nouns (genitives), between nouns and their modifiers (adjectival modification, relativization), between verbs and their modifiers (adverbial modification including clausal subordination). A detailed dependency label tagset encompassing all these different kinds of relations is defined in the annotation scheme based on the CPG formalism [23].

One important point of departure in CPG from other dependency grammars is that dependency relations may also be defined between groups of words known as *chunks*. A chunk is defined as a minimal, non-recursive structure consisting of a group of related words. Each chunk has a unique head word whose category determines the chunk type. This head word is modified by the other words in the chunk. In a chunk-based dependency representation, dependency relations are defined between chunk heads. Another important concept in CPG which relates to the notion of a chunk is the *vibhakti*. For a noun chunk, *vibhakti* is the post-position/suffixes occurring after the noun which encodes information about case-marking and thematic roles (via the notion of *karaka* which is syntactico-semantic). Similarly, in the case of a verb chunk, the verbal head may be followed by auxiliary verbs which may remain as separate words or combine with the head as suffixes depending on the morphology of the language. This information following the head is collectively called the *vibhakti* of the verb. The *vibhakti* of a verb chunk encodes information about the tense, aspect and modality (TAM) as well as agreement features of the verb. Both these kinds of *vibhakti* have been shown to be crucial in NLP applications such as parsing [24]. In fact, even during annotation, nominal *vibhaktis* serve as cues to identify the *karaka* relation that can be assigned to the noun.

IV. External Sandhi and its Relevance to Syntactic Annotation

The origins of the notion of *sandhi* can be traced back to the seminal work of theorists such as Paṇini in the Indian linguistic tradition. Briefly stated, sandhi ('putting together') refers to a set of morpho-phonological processes that occur at either morpheme or word boundaries. These processes are captured as sandhi rules in traditional Sanskrit grammars which are based chiefly on the **avoidance of hiatus** and on **assimilation** [25]. It must be noted that sandhi is also reflected in the orthography of Sanskrit as the coalescence of final and initial letters. Two types of sandhi are identified in language, **internal sandhi** and **external sandhi**. Internal sandhi refers to word-internal morphonological changes that take place at morpheme boundaries during the process of word-formation. The internal sandhi rules in Sanskrit grammar apply to the final letters of verbal roots and nominal stems when followed by certain suffixes or terminations [25]. An example of internal sandhi in English would be the positional variation of the negative morpheme 'in-' to give the allomorph 'im-' when it is prefixed to words that begin with bilabial sounds (as in 'impossible'). Such processes lie obviously, within the domain of morphology. External sandhi, on the other hand, refers to processes that apply word-externally (across *word* boundaries). External sandhi rules in Sanskrit grammar determine the changes of final and initial letters of words [25]. Examples of external sandhi formation in English are the well-known cases of *wanna*/*hafta*/*gotta* contractions where the verb combines with the infinitival 'to' following it to give the contracted form. Note that external sandhi as seen in these examples need not always be reflected orthographically in English ('want to' while writing, but spoken as 'wanna').

The notion of sandhi formation is also well-known in modern linguistics. However, in much of the literature on this subject, sandhi is treated purely as a phonological process. That phonological processes occur across morpheme boundaries and not across word boundaries is the default situation in much of phonology. Cases where phonological rules apply across word boundaries, in other words, cases of external sandhi formation, have attracted special attention in generative phonology. In fact, this phenomenon is one of the central motivations for the theory of *Prosodic Phonology* [26], which formalizes the intervention of syntactic conditions (the relationships between words) in phonological matters. A similar notion discussed in the literature is that of *phonological phrase* which is defined as an organizational unit in phonology (parallel to syntactic phrase in syntax). A phonological phrase, therefore, is the domain within which external sandhi rules operate [27].

As discussed earlier in the case of English contractions, external sandhi can be limited to connected speech and need not be present in text. However, in Indian languages, especially Dravidian languages, sandhi (both internal and external) has a wide-spread occurrence and is also orthographically reflected most of the time. Sandhi phenomena in European languages have also been studied extensively [28] and external sandhi has been discussed as occurring prominently in Italian [29]. We refer to all such languages with high prevalence of external sandhi as Sandhi languages. External sandhi formation in

Sandhi languages leads to fusion of morphological words resulting in morphologically complex/unanalyzable forms. This poses a problem for all natural language processing applications such as POS-tagging, chunking, parsing, etc. that deal with written text. The task of tokenization becomes complex in these languages as tokens obtained through sentence splitting can contain more than one morphological word within them. Since external sandhi is a consequence of (orthographically visible) phonological processes occurring at the prosodic level, splitting such instances of sandhi can not fall within the purview of the morph analyzer. The task of splitting sandhi forms requires segmentation at a different level and should be treated as being distinct from morphological segmentation. Without this distinction between sandhi formation and other kinds of morphological changes, the task of morphological analysis in languages like Telugu becomes extremely complex. In the case of syntactic treebanking, if cases of external sandhi are not handled appropriately during tokenization, information about the syntactic relations that obtain between the words fused together due to external sandhi formation would be lost. This can be seen from the following Telugu examples.

```
(kanpu)_NP (warvAwa)_NP (wagina)_VGNF
childbirth after        appropriate
    (saMrakRaNa)_NP (lexanukoMdi)_VGF
    protection      Neg-Fin+think-Fin-Sandhi
'Think that there is no proper protection
    post-partum'
```

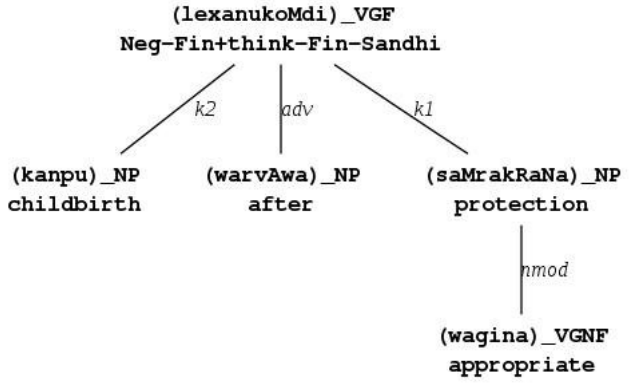

Fig. 1. Example dependency tree from the Telugu treebank without sandhi splitting.

In this example, the verb in the matrix clause is 'think' (Telugu stem: 'anukoVtam') which takes a clausal complement. The verb in the complement clause is a Negative finite verb ('lexu'). It can be noticed that in the above example, both the verbs are fused together as a result of external sandhi formation. If this issue is not handled prior to dependency annotation, dependencies would have to be incorrectly shown between the chunks in the sentence and the fused form ('lexanukoMdi') which is treated as the head of the sentence. This is linguistically inappropriate for the obvious reason that the information pertaining to the argument structure of both the verbs in this sentence is lost in such a representation. The dependency tree corresponding to this sentence from version 0.2 in Fig. 1 clearly shows this fallacy. The correct dependency tree for this sentence is also shown in Fig. 2 for comparison.

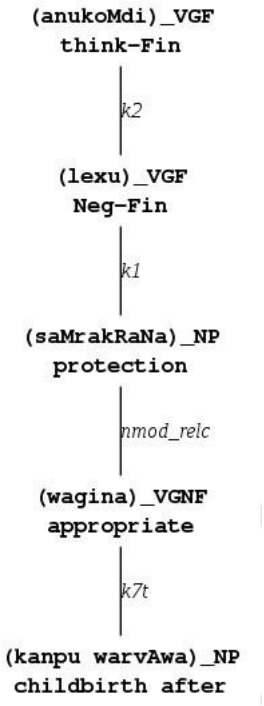

Fig. 2. Example dependency tree from the Telugu treebank with sandhi splitting.

The following example sentences too contain instances of external sandhi formation.

```
(rogaM)_NP (muxirina)_VGNF (pillani)_NP
disease     worsen         girl-Acc
    (xavAKAnAlo)_NP (cuparemi)_VGF
    hospital-Loc    show-Fin-emi-Sandhi
'Why dont (you) show this disease afflicted
    girl in a hospital?'

(ammakemi)_NP (wocakuMdeV)_VGF
Mother-Dat-emi-Sandhi strike-Fin
'Mother could not think of anything'
(literal: Anything did not strike mother)
```

The word 'emi' undergoes external sandhi in both these sentences. While in the first sentence, it gets fused to the predicate in that clause ('cuparu'), in the second sentence, it fuses with another argument ('ammaku') of the predicate. Corresponding to this difference in sandhi formation, there exists an interesting difference in the function of 'emi' which can be noticed from the translations provided. In the first sentence where it fuses with the verb, its function is similar to that of a question word (or clitic). In the second sentence, however, it functionally resembles a Negative Polarity Item (NPI). This correlation between sandhi formation and the

function of 'emi' can be seen as a direct evidence for the interaction between syntax and prosody. While syntactically annotating these sentences, if the sandhi form is not split, information about the syntactic relation of 'emi' to its head would be lost. In fact, this word/clitic 'emi' belongs to a paradigm of demonstrative pronouns in Telugu all of which can potentially exhibit similar behaviour. If the sandhi in each of these sentences is not split, syntactically important information such as argument structure would be either lost or misrepresented in the treebank.

In the next section, we describe how such sandhi forms in the previous version of the treebank were manually split leading to a new version of the treebank.

V. Sandhi Splitting in Syntactic Treebanking

The examples in the previous section show how external sandhi formation in Telugu, can lead to either loss or misrepresentation of syntactic information in the treebank. The sandhi forms in such sentences need to be split so that syntactic relationships involving tokens undergoing sandhi are accurately represented. In this section, we discuss the rationale for introduction of a distinct sandhi splitting stage in the annotation pipeline. We also describe how sandhi forms in the Telugu treebank were split to produce a new sandhi split version. However, we first give a brief overview of the annotation pipeline being followed for IL treebanking.

A. Annotation Pipeline for IL Treebanking

The annotation process followed to develop a treebank of CPG-based dependency structures for Indian languages consists of multiple steps (see Fig. 3). Sentences are tokenized to begin with. The tokens obtained at the end of this step are analyzed by a morph analyzer in the next step. At the third stage, the tokens in the sentence are POS-tagged which is followed by chunking at the fourth stage where tokens are grouped into chunks. As mentioned earlier, it is possible in this annotation scheme to annotate dependency relations between chunks (in fact, *heads* of chunks). This distinction between inter-chunk and intra-chunk dependencies is based on the observation that intra-chunk dependencies can be generated with high accuracy given the chunks and their heads (except in very few cases such as compounds, collocations). Thus, annotation of inter-chunk dependencies alone in phase 1 of the annotation would result in a chunk-level dependency treebank. This strategy also minimizes the time requirements of syntactic treebank development which is usually seen as a labor-intensive and time-consuming task. At the next stage in the pipeline, the dependency relations are annotated between the chunks. This is followed by post-processing in the form of quality checks and validation. The processing at each of these stages can be automated followed by human post-editing. Information obtained at each stage of processing is used by subsequent stages. Currently, processing at the first four stages, namely tokenization, morph analysis, POS-tagging and chunking, is being reliably done using highly accurate tools. The task of dependency annotation can also be automated using *vibhaktis* as cues for *karaka* assignment. However, it must be noted that *vibhaktis* can also be ambiguous which is why the task of *karaka* assignment is not always straight-forward. Therefore, it was decided that reliable annotation of syntactic dependencies can be achieved only through manual annotation.

In order to get an idea about the degree of occurrence of external sandhi, and also about the different kinds of sandhi possible, a detailed manual study of 600 sentences from the previous version of the Telugu treebank was done. We observed that there was no straight-forward method to identify sandhi forms in Telugu. The assumption that sandhi forms can be identified based on the output of the morphological analyzer is not correct. This is because forms for which the paradigm-based morph analyzer does not generate any analysis include, apart from cases of external sandhi, inflections of unknown words. In addition, the morph analyzer sometimes analyses sandhi forms incorrectly treating them as words. These observations suggest that splitting of sandhi forms should precede morph analysis. In fact, Sandhi splitting must be done as part of the tokenization step as external sandhi causes the fusion of tokens. Once the tokens in a sentence are obtained, all the other steps in the annotation process can be carried out without any changes. Fig. 3 shows the annotation pipeline with a sandhi splitting stage introduced prior to morph analysis. This additional stage would be necessary for all Sandhi languages.

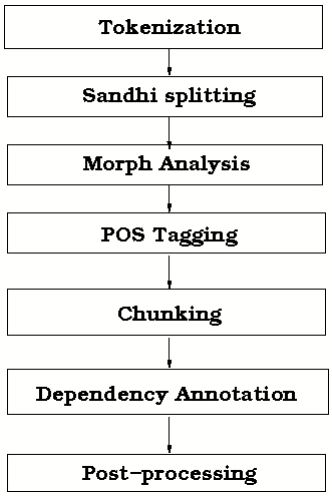

Fig. 3. Modified annotation pipeline for Sandhi languages such as Telugu.

B. Sandhi Splitting in the Telugu Treebank

In the course of the manual study, it was observed that *e*-demonstratives in Telugu (such as 'emi' in the examples from the previous section), undergo sandhi with neighbouring words most of the time. In this regard, they can be compared

TABLE II
DIFFERENT KINDS OF MODIFICATIONS MADE TO VERSION 0.2 OF THE TELUGU TREEBANK

Type of modification	# of modifications
POS-tag corrections	239
Chunk tag corrections	136
DepRel corrections	673
Sandhi splitting changes	179
All	1227

to what are known an *leaners* such as 'to' in English [27]. Another interesting observation about these elements is that their function seems to vary according to the part of speech of the word they lean on. Since this class of pronouns has a distinct form, it is possible to easily extract their instances from the treebank. All instances of external sandhi involving these demonstratives were extracted and split. Since this work is a preliminary exploration of external sandhi in the Telugu treebank, we restrict ourselves to manual sandhi splitting. However, in order to be able to split all types of external sandhi over the entire treebank, automating the task of sandhi segmentation is a mandatory requirement. It is expected that the sandhi rules in Telugu we encountered during the process of manual sandhi splitting would aid in the development of an automatic segmentizer.

In the process of manual sandhi splitting from the previous version of the treebank, we encountered POS-tag and chunk tag errors made by the automatic taggers which were corrected. Errors in annotation of syntactic dependencies (both attachment and relation label) from earlier phases of annotation were also corrected. The statistics about all these different kinds of modifications are given in Table II. The new version of the treebank resulting from these modifications is numbered as version 0.5[7].

VI. PARSING EXPERIMENT

Although treebanks can be used for a variety of purposes, the major impetus for treebanking in recent times, has come from the rapid developments in the area of data-driven natural language parsing. In fact, the relationship of sandhi formation with the syntax of Telugu discussed in this work was discovered in the light of a detailed analysis of the results of the NLP tools' contest for IL parsing at ICON 2009[8]. In order to empirically verify the usefulness of sandhi splitting in the treebank for syntactic parsing, we experiment by applying a data-driven dependency parser first, to sentences from the previous version of the treebank and then, to the new version in which sandhi splitting was manually done. A comparison of the parsing accuracies obtained using these two versions of the treebank would help us understand not only the effect of sandhi formation on Telugu parsing but also the efficiency of the design choices we made to address it in the new

[7]This new version of the treebank is released for the NLP tools' contest on IL parsing at ICON 2010. *http://ltrc.iiit.ac.in/nlptools2010/*

[8]*http://ltrc.iiit.ac.in/nlptools2009/*

TABLE III
DESCRIPTION OF THE DATASETS USED IN THE PARSING EXPERIMENTS

Dataset	Description
set-0	1600 sentences from treebank version 0.2
set-1	POS-tag, Chunk tag and DepRel error corrections
set-2	sandhi-splitting changes only
set-3	both changes

version of the treebank. In addition, as already mentioned in the previous section, modifications made to the previous version of the treebank include post-editing changes wherein the errors of the automatic POS-tagger and chunker are corrected and also, dependency corrections (both attachment and label corrections). For our parsing experiments, we created four different datasets each containing a different version of the same set of sentences. Set-0 contains sentences drawn from the previous version of the treebank. The sentences in set-0 are replaced by their post-edited (POS-tag, chunk tag and dependency relation corrected) versions to create set-1. Sentences in set-0 containing instances of external sandhi are replaced by their sandhi split versions to create set-2. Finally, set-3 is made up of sentences containing both post-editing and sandhi-splitting changes. The details of the datasets are briefly summarized in Table III.

Applying a data-driven parser to these different datasets, we tried to tease apart the influence of these different kinds of modifications on the parsing accuracy. We use the publicly available MaltParser [30] in this experiment with learner and feature model settings identical to those of the system that reported the highest accuracies for Telugu parsing at the NLP tools' contest 2009. In order to be able to pin-point the effect of the annotation changes on the parser performance and also, to normalize for sentence length and complexity, we ran the parser in cross-validation mode (10-fold) besides applying it to a test set of 150 sentences. Both these accuracies for each dataset are shown in table IV.

As shown in table III, set-0 is comprised of sentences drawn from the previous version of the treebank. Therefore, the accuracies obtained on set-0 are treated as the baseline accuracies in this experiment. It must be noted that the baseline accuracies obtained in our experiment using set-0 are considerably higher than the best accuracies reported on the same version of the treebank released for the NLP tools' contest shared task on parsing 2009 [18]. This difference in accuracies can be attributed solely to the way the treebank was partitioned to create the released datasets. The accuracies obtained on set-1 are slightly greater than the baseline accuracies. The increase in unlabeled attachment score (UAS) (both cross-validation and test set) is higher than the one in labeled attachment score (LAS). This difference between the accuracies obtained on set-1 and the baseline accuracies demonstrates the effect of correction of errors from the previous version of the treebank.

The accuracies obtained on set-2, although better than the baseline accuracies, are less than the accuracies obtained

TABLE IV
ACCURACIES ON THE FINE-GRAINED DATASETS

Dataset	Cross-Validation			Test Set		
	LAS	UAS	LS	LAS	UAS	LS
set-0	67.45	87.85	70.37	66.90	87.18	70.02
set-1	67.96	88.96	70.57	67.65	88.06	70.59
set-2	67.77	87.94	70.66	67.06	88.63	69.40
set-3	68.31	89.50	70.37	68.28	88.98	70.45

TABLE V
ACCURACIES ON THE COARSE-GRAINED DATASETS

Dataset	Cross-Validation			Test Set		
	LAS	UAS	LS	LAS	UAS	LS
set-0	71.87	87.97	75.37	69.32	88.91	72.10
set-1	73.24	89.55	75.78	71.80	90.14	74.22
set-2	72.20	88.32	75.33	69.73	88.29	72.07
set-3	73.61	89.86	75.89	71.29	89.98	73.29

on set-1. The accuracies obtained on this set reflect the effect of splitting just one type of external sandhi. This is understandable given that the number of sandhi splitting changes in the treebank is much less than the error corrections made to create set-1 (see table II). In the cross-validation experiments with set-1, we observed that the performance of the parser improves as the number of folds is increased. This suggests that the parser needs more training data to learn the new structures created as a result of sandhi splitting in the treebank.

The performance of the parser on set-3 is significantly better than both set-1 and set-2. The increase is significant in both LAS and UAS. This shows that post-editing changes such as POS-tag and chunk tag corrections as well as dependency corrections also aid in the learning of sandhi split structures. The improvement in UAS (1.65 for cross-validation and 1.80 on the test set) is more than that of LAS (0.86 for cross-validation and 1.38 on the test set).

We also repeated this experiment with similar datasets created using coarse-grained data [9]. The parsing accuracies on the coarse-grained datasets are shown in Table V. The results of this experiment exhibit a trend similar to that observed in the case of fine-grained data. However, it must be noted that the increase in LAS is expectedly much higher in the case of coarse-grained data. Overall, the results justify our claim about the importance of splitting sandhi forms in the treebank for the task of NL parsing.

VII. CONCLUSIONS

In this paper, we introduced the linguistic phenomenon of external sandhi in Telugu, an Indian language. We discuss how external sandhi formation in Telugu poses a problem in the syntactic annotation of Telugu sentences. We show using examples, that external sandhi, if not handled prior to dependency annotation in the treebanking process, can lead to either loss or misrepresentation of syntactic information. We report the insights gained from a detailed study of the instances of external sandhi from version 0.2 of the Telugu treebank. Based on these insights, we propose a modification to the generic annotation pipeline which would be relevant for all Sandhi languages. We manually split instances of one type of external sandhi widely occurring in the previous version of the treebank. In addition to sandhi-splitting, post-editing changes which include POS-tag corrections, chunk tag corrections and dependency (both attachment and label) corrections were also carried out, resulting in the development of a new version 0.5 of the Telugu treebank. Finally, we conduct an experiment with a statistical parser to empirically verify the usefulness of sandhi-splitting for the NL parsing task. The results of our experiment show that splitting instances of even just one type of external sandhi has a salubrious effect on the overall parsing accuracies. Developing an automatic sandhi-segmenter for Telugu based on our experience of manual sandhi-splitting is part of our immediate future work.

[9]The number of distinct dependency labels in the fine-grained data (44) is reduced to 22 coarse-grained labels.

ACKNOWLEDGMENTS

The authors would like to thank Viswanatha Naidu, Samar Husain, Prashanth Mannem, Sukhada and Sriram Venkatapathy for the helpful comments and discussions. Thanks are also due to all the annotators who did the initial annotation. The authors especially appreciate Viswanatha Naidu's sterling efforts that led to the development of HyDT-Telugu 0.2.

REFERENCES

[1] M. Marcus, M. Marcinkiewicz, and B. Santorini, "Building a large annotated corpus of English: The Penn Treebank," *Computational linguistics*, vol. 19, no. 2, pp. 313–330, 1993.

[2] A. Bharati, V. Chaitanya, and R. Sangal, *Natural language processing: a Paninian perspective*. Prentice Hall of India, 1995.

[3] J. Hajič, "Building a Syntactically Annotated Corpus: The Prague Dependency Treebank," *Issues of Valency and Meaning. Studies in Honour of Jarmila Panevová*, 1998.

[4] E. Hajicová, "Prague Dependency Treebank: From Analytic to Tectogrammatical Annotation," *Proceedings of TSD'98*, pp. 45–50, 1998.

[5] K. Oflazer, B. Say, D. Hakkani-Tür, and G. Tür, "Building a Turkish treebank," *Treebanks: Building and Using Parsed Corpora*, vol. 20, pp. 261–277, 2003.

[6] A. Culotta and J. Sorensen, "Dependency tree kernels for relation extraction," in *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2004, p. 423.

[7] F. Reichartz, H. Korte, and G. Paass, "Dependency tree kernels for relation extraction from natural language text," *Machine Learning and Knowledge Discovery in Databases*, pp. 270–285, 2009.

[8] R. Begum, S. Husain, A. Dhwaj, D. Sharma, L. Bai, and R. Sangal, "Dependency annotation scheme for Indian languages," *Proceedings of IJCNLP-2008*, 2008.

[9] R. Bhatt, B. Narasimhan, M. Palmer, O. Rambow, D. Sharma, and F. Xia, "A multi-representational and multi-layered treebank for hindi/urdu," in *Proceedings of the Third Linguistic Annotation Workshop*. Association for Computational Linguistics, 2009, pp. 186–189.

[10] M. Palmer, R. Bhatt, B. Narasimhan, O. Rambow, D. Sharma, and F. Xia, "Hindi Syntax: Annotating Dependency, Lexical Predicate-Argument Structure, and Phrase Structure," in *The 7th International Conference on Natural Language Processing*, 2009, pp. 14–17.

[11] A. Bhatia, R. Bhatt, B. Narasimhan, M. Palmer, O. Rambow, D. Sharma, M. Tepper, A. Vaidya, and F. Xia, "Empty Categories in a Hindi Treebank," in *LREC-2010*, 2010.

[12] A. Bharati, M. Bhatia, V. Chaitanya, and R. Sangal, "Paninian Grammar Framework Applied to English," *South Asian Language Review*, 1997.

[13] A. Vaidya, S. Husain, P. Mannem, and D. Sharma, "A Karaka Based Annotation Scheme for English," *Computational Linguistics and Intelligent Text Processing*, pp. 41–52, 2009.

[14] C. Vempaty, V. Naidu, S. Husain, R. Kiran, L. Bai, D. Sharma, and R. Sangal, "Issues in Analyzing Telugu Sentences towards Building a Telugu Treebank," *Computational Linguistics and Intelligent Text Processing*, pp. 50–59, 2010.

[15] B. Krishnamurti, *The Dravidian languages*. Cambridge Univ Press, 2003.

[16] E. Sapir, *Language: An introduction to the study of speech*. Dover Publications, 1921.

[17] J. Greenberg, "A quantitative approach to the morphological typology of language," *International Journal of American Linguistics*, vol. 26, no. 3, pp. 178–194, 1960.

[18] S. Husain, "Dependency Parsers for Indian Languages," *Proceedings of ICON09 NLP Tools Contest: Indian Language Dependency Parsing*, 2009.

[19] A. Bharati, R. Sangal, D. Sharma, and L. Bai, "Annotating corpora guidelines for pos and chunk annotation for indian languages," 2006, technical report: TR-LTRC-31, LTRC.

[20] S. Buchholz and E. Marsi, "CoNLL-X shared task on multilingual dependency parsing," in *Proceedings of the Tenth Conference on Computational Natural Language Learning*. Association for Computational Linguistics, 2006, pp. 149–164.

[21] 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*. Association for Computational Linguistics, 2007.

[22] V. Mittal, "Automatic Sanskrit segmentizer using finite state transducers," in *Proceedings of the ACL 2010 Student Research Workshop*. Association for Computational Linguistics, 2010, pp. 85–90.

[23] A. Bharati, D. Sharma, S. Husain, L. Bai, R. Begum, and R. Sangal, "AnnCorra: Treebanks for Indian languages, guidelines for annotating Hindi dependency treebank," 2009, http: // ltrc.iiit.ac.in / MachineTrans / research/tb/DS-guidelines/DS-guidelines-ver2-28-05-09.pdf.

[24] B. Ambati, S. Husain, J. Nivre, and R. Sangal, "On the role of morphosyntactic features in Hindi dependency parsing," in *The First Workshop on Statistical Parsing of Morphologically Rich Languages (SPMRL 2010)*, 2010, pp. 94–102.

[25] A. A. Macdonell, *A Sanskrit Grammar for students*. New Delhi, India: D.K. Printworld (P) Ltd., 1926.

[26] E. Selkirk, "On prosodic structure and its relation to syntactic structure," *Nordic Prosody II: Papers from a Symposium*, pp. 111–140, 1981.

[27] A. Zwicky, "Stranded *to* and phonological phrasing in english," *Linguistics*, vol. 20, pp. 3–57, 1982.

[28] H. Andersen, *Sandhi phenomena in the languages of Europe*. Mouton de Gruyter, 1986.

[29] M. Absalom and J. Hajek, "Prosodic phonology and raddoppiamento sintattico: a re-evaluation," in *Selected Papers from the 2005 Conference of the Australian Linguistic Society, Melbourne: Monash University. http://www. arts. monash. edu. au/ling/als*, 2006.

[30] 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*, vol. 13, no. 02, pp. 95–135, 2007.